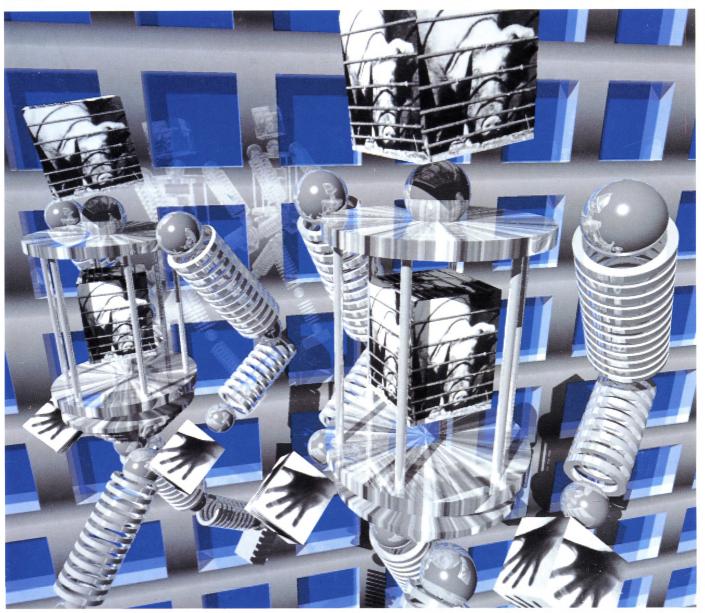
1990年 9 月 1 日発行（毎月 1 回 1 日発行）第 9 巻 9 号通巻101号　昭和58年11月 2 日第三種郵便物認可

PERSONAL COMPUTER MAGAZINE for MZ, X1, and X68000

特集1 日本語を処理するための序章

特集2 ADVANCED 2D GRAPHICS続論
清水和人流「プログラミング道場」
X68000ハンディスキャナの接続

1990

オー！エックス
定価560円

ひらかれた知性。

ザ・ワークステーション。80Mバイト(SCSI仕様)ハードディスク、SCSIインターフェイスを標準装備。

SUPER HD

本体＋キーボード＋マウス・トラックボール

CZ-623C-TN(チタンブラック) 標準価格498,000円(税別)

アートの系譜。

EXPERT II

本体＋キーボード＋マウス・トラックボール

CZ-603C-BK(ブラック)・-GY(グレー) 標準価格338,000円(税別)/HDタイプ CZ-613C-BK(ブラック) 標準価格448,000円(税別)

ニュースタンダード。

PRO II

本体＋キーボード＋マウス

CZ-653C-BK(ブラック)・-GY(グレー) 標準価格285,000円(税別)

HDタイプ CZ-663C-BK(ブラック)・-GY(グレー) 標準価格395,000円(税別)

資料請求券
X68000
Oh!X
9月

次代のユーザーインターフェイスを象徴する"SX-WINDOW※"搭載。

今回のX68000ニューシリーズのデビューに関して、ハードウェア以上にウィンドウ環境の提供に耳目が集中したことは、昨今のビジュアルユーザーインターフェイス事情をふまえれば、当然のことと言えるでしょう。マルチウィンドウを駆使してX68000をコントロールする、待ち望まれていた環境がこのSX-WINDOWによって実現されるのです。何の予備知識もなしにこのウィンドウに接した方は、一見して従来のビジュアルシェルのバージョンアップと思われるかもしれませんが、本質的には全く異質のものと言えます。ひとつのウィンドウである仕事をさせながら、別のウィンドウで違う仕事にとりかかる。ひとことで言えばアプリケーションを実行させる環境としてのウィンドウであるということ。これまでのビジュアルシェルではできなかったシーンを生み出しています。複数のアプリケーションを同じ操作のもとで走らせたり、アプリケーション相互でデータのやりとりが可能になるわけです。そして、次代のインテリジェンスを鮮やかに象徴する4階調のハイセンスな画面処理——。SX-WINDOWをターゲットとしたアプリケーション開発もすでに推進されており、これからの展望という点からも大いに期待されるところです。また、このSX-WINDOWはディスクによって供給され、BIOSの高速化(平均2倍)も含めてOSであるHuman68kの機能を拡張。ニューシリーズのみならず、すべてのX68000でこの新しい環境が享受できます。

※SX-WINDOWの起動には、メインメモリ2MBが必要です。CZ-600C/601C/611C/652C/653C/662C/663CでSX-WINDOWをご使用の際には、あらかじめ別売の1MB増設RAMボードを増設してください。

NEW

X68000

PERSONAL WORKSTATION

SUPER・EXPERT・PRO

充実のディスプレイラインアップ

機種	型番	価格
15型カラーディスプレイテレビ(ドットピッチ0.39mm)	CZ-602D-BK(ブラック)・-GY(グレー※)	標準価格 99,800円(チルトスタンド同梱・税別)
15型カラーディスプレイテレビ(ドットピッチ0.39mm)	CZ-605D-BK(ブラック)・-GY(グレー)	標準価格115,000円(スピーカー2個/チルトスタンド同梱・税別)
15型カラーディスプレイテレビ(ドットピッチ0.31mm)	CZ-613D-TN(チタンブラック)・-BK(ブラック)・-GY(グレー)	標準価格135,000円(スピーカー2個/チルトスタンド同梱・税別)
14型カラーディスプレイ(ドットピッチ0.31mm)	CZ-603D-BK(ブラック)・-GY(グレー)	標準価格 84,800円(チルトスタンド同梱・税別)
14型カラーディスプレイ(ドットピッチ0.31mm)	CZ-604D-BK(ブラック)・-GY(グレー)	標準価格 94,800円(スピーカー2個/チルトスタンド同梱・税別)
21型カラーディスプレイ(ドットピッチ0.52mm)	CU-21HD-BK(ブラック)	標準価格148,000円(スピーカー2個同梱・税別)

※印の商品は在庫僅少です。

シャープX68000
パソコン教室開催中

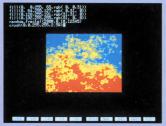
●会場:市ヶ谷教室 シャープ東京支社ビル●コース:入門コース・表集計コース・音楽コース・絵画コース●申込受付電話番号:(03)260-8365●受講料:2,000円(税別)

EXEリーダーズグッズ
プレゼント実施中

●いま、EXE会員よりご紹介のお客様がEXEショップでX68000シリーズを購入されますと、EXE会員にEXEリーダーズグッズをプレゼントします。詳しくはEXEショップにお問い合わせください。●また、X68000シリーズをご購入のお客様は、ぜひEXEクラブにご入会ください。

●お問い合わせは…シャープ(株)電子機器事業本部システム機器営業部 〒545大阪市阿倍野区長池町22番22号☎(06)621-1221(大代表)
電子機器事業本部液晶映像システム事業部第2商品企画部 〒162東京都新宿区市谷八幡町8番地☎(03)260-1161(大代表)

シャープ株式会社

ハンディイメージスキャナアダプタの製作

特集2　ADVANCED 2D GRAPHICS続論

BILLLIARDS

赤黒（SPEED）

トンネルズ&トロールズ

シムシティー

Oh!X

C O N T

〈スタッフ〉
●編集長／前田　徹　●編集／植木章夫　岡崎栄子　浅井研二　●協力／有田隆也　中森　章　後藤貴行　林　一樹　荻窪　圭　岡本浩一郎　毛内俊行　吉田賢司　影山裕昭　相馬英智　古村　聡　村田敏幸　丹　明彦　三沢和彦　長沢淳博　宮島　靖　金子俊一　浦川博之　山田純二　●カメラ／杉山和美　●イラスト／永沢しげる　山田晴久　小栗由香　●アートディレクター／島村勝頼　●レイアウト／元木昌子　AD GREEN　●校正／グループごじら

表紙絵：塚田　哲也

1990 SEP.
9

E　N　T　S

●読みもの

■広告目次

クリエイティブマインドあふれる周辺機器が

ディスプレイ関連

カラーディスプレイテレビ

15型カラーディスプレイテレビ
CZ-602D-BK
★CZ-602D-GY
標準価格 99,800円(税別)
(チルトスタンド同梱)

15型カラーディスプレイテレビ
CZ-605D-BK・-GY
標準価格 115,000円(税別)
(スピーカー2個・チルトスタンド同梱)

15型カラーディスプレイテレビ
CZ-613D-TN・-BK・-GY
標準価格 135,000円(税別)
(スピーカー2個・チルトスタンド同梱)

カラーディスプレイ

14型カラーディスプレイ
CZ-603D-BK・-GY
標準価格 84,800円(税別)
(チルトスタンド同梱)

14型カラーディスプレイ
CZ-604D-BK・-GY
標準価格 94,800円(税別)
(スピーカー2個・チルトスタンド同梱)

21型カラーディスプレイ
CU-21HD
標準価格 148,000円(税別)
(スピーカー2個同梱)

CRTフィルター

高性能CRTフィルター
BF-68PRO
標準価格 19,800円(税別)
(14/15型用)

チューナー

RGBシステムチューナー
CZ-6TU-BK・-GY
標準価格 33,100円(税別)
(リモコン付)

アートツール

画像入力

カラーイメージスキャナ※1
CZ-8NS1
標準価格 188,000円(税別)

スキャナ用パラレルボード
CZ-6BN1
標準価格 29,800円(税別)

映像入力

カラーイメージユニット※2
CZ-6VT1-BK
CZ-6VT1
標準価格 69,800円(税別)

映像出力

ビデオボード
CZ-6BV1
標準価格 21,000円(税別)
※拡張I/Oポートを2スロット使用します。

プリンタ

カラープリンタ

24ドット
熱転写カラー漢字プリンタ
★CZ-8PC3
標準価格 65,800円(税別)
(信号ケーブル同梱)

48ドット
熱転写カラー漢字プリンタ
CZ-8PC4
CZ-8PC4-GY
標準価格 99,800円(税別)
(信号ケーブル同梱)

カラービデオプリンタ

カラービデオプリンタ
CZ-6PV1
標準価格 198,000円(税別)
(信号ケーブル同梱)

カラーイメージジェット

カラーイメージジェット※3
IO-735X
標準価格 248,000円(税別)
(信号ケーブル別売)

ドットプリンタ

24ピン
カラー漢字プリンタ(80桁)
CZ-8PG1
標準価格 130,000円(税別)
(信号ケーブル同梱)

24ピン
カラー漢字プリンタ(136桁)
CZ-8PG2
標準価格 160,000円(税別)
(信号ケーブル同梱)

24ピン漢字プリンタ(136桁)
CZ-8PK10
標準価格 97,800円(税別)
(信号ケーブル同梱)

ファイル

ハードディスク

ハードディスクユニット(20MB)
CZ-620H
標準価格 178,000円(税別)

増設用ハードディスク
ドライブ (40MB)
(CZ-602C/603C/652C
653C内蔵用)
CZ-64H
標準価格 120,000円(税別)
(取付費別)
※取付に関してはシャープ
お客様ご相談窓口にてご
相談ください。

※1 ご使用に際しては、カラーイメージスキャナCZ-8NS1に同梱のRS-232Cケーブルで接続するか、より高速のパラレルデータ伝送を行う場合、別売のスキャナ用パラレルボードCZ-6BN1標準価格29,800円(税別)で接続してください。
※2 CZ-603D/604D、CU-21HDをご使用の場合は、RGBシステムチューナーCZ-6TU(別売)が必要です。
※3 別売の信号ケーブルIO-73CX標準価格5,500円(税別)で接続して下さい。

XVI・XVI turbo シリーズ用 周辺機器
標準価格は税別です。

カラーディスプレイ		
●21型カラーディスプレイ※1	CU-21HD	148,000円

映像・画像入力編集装置		
●カラーイメージスキャナ	CZ-8NS1	188,000円
●カラーイメージボードII	CZ-8BV2	39,800円

●立体映像セット	★CZ-8BR1	29,800円
●パーソナルテロッパ※2	CZ-8DT2	44,800円

FM音源		
●ステレオタイプFM音源ボード	CZ-8BS1	23,800円

スピーカー(2本1組)標準装備、ミュージックツール同梱

プリンタ		
●24ピンカラー漢字プリンタ(80桁)	CZ-8PG1	130,000円
●24ピンカラー漢字プリンタ(136桁)	CZ-8PG2	160,000円

●24ピン漢字プリンタ(136桁)	CZ-8PK10	97,800円
●24ドット熱転写カラー漢字プリンタ	★CZ-8PC3	65,800円
●48ドット熱転写カラー漢字プリンタ	CZ-8PC4	99,800円
●48ドット熱転写カラー漢字プリンタ	CZ-8PC4-GY	99,800円
●カラービデオプリンタ	CZ-6PV1	198,000円
●カラーイメージジェット	IO-735X	248,000円

ファイル		
●ミニフロッピーディスクユニット(2HD・2D)※3	★CZ-520F	118,000円

シャープ株式会社 ●お問い合わせは…シャープ㈱電子機器事業本部システム機器営業部 〒545 大阪市阿倍野区長池町22番22号☎(06)621-1221(大代

X68000をサポート。

シャープペリフェラルファミリー X68000

ボード

拡張メモリ

1MB増設RAMボード
(CZ-600C専用)
CZ-6BE1
標準価格 35,000円(税別)

1MB増設RAMボード
(CZ-601C/611C/652C/653C/662C/663C用)
CZ-6BE1B
標準価格 28,000円(税別)

2MB増設RAMボード ※4
CZ-6BE2
標準価格 79,800円(税別)

4MB増設RAMボード ※4
CZ-6BE4
標準価格 138,000円(税別)

インターフェイス

ユニバーサルI/Oボード
CZ-6BU1
標準価格 39,800円(税別)

GP-IBボード
CZ-6BG1
標準価格 59,800円(税別)

増設用RS-232Cボード
(2チャンネル)
CZ-6BF1
標準価格 49,800円(税別)

数値演算プロセッサ

数値演算プロセッサボード
CZ-6BP1
標準価格 79,800円(税別)

FAX

FAXボード
CZ-6BC1
標準価格 79,800円(税別)

MIDI

MIDIボード
CZ-6BM1
標準価格 26,800円(税別)

ネットワーク

モデム

モデムユニット ※5
CZ-8TM2
標準価格 49,800円(税別)
(RS-232Cケーブル同梱)

RS-232Cケーブル

RS-232Cケーブル
(平行接続型)
CZ-8LM1
標準価格 7,200円(税別)

RS-232Cケーブル
(クロス接続型)
CZ-8LM2
標準価格 7,200円(税別)

LANボード

LANボード NEW
CZ-6BL1
標準価格268,000円(税別)
(イーサネット用)
CZ-6BL2
標準価格298,000円(税別)
(イーサネット/チーパネット両用)
※電源ユニット・ソフトウェア
(ネットワークドライバVer1.0)同梱

入力

インテリジェントコントローラ
CZ-8NJ2
標準価格 23,800円(税別)

マウス・トラックボール
CZ-8NM3
標準価格9,800円(税別)

トラックボール
CZ-8NT1
標準価格 13,800円(税別)

マウス
CZ-8NM2A
標準価格 6,800円(税別)

ジョイカード
CZ-8NJ1
標準価格 1,700円(税別)

その他

拡張スロット

拡張 I/Oボックス(4スロット)
(CZ-600C/601C/602C/603C/611C/612C/613C/623C用)
CZ-6EB1-BK
CZ-6EB1
標準価格 88,000円(税別)

スピーカー

アンプ内蔵
スピーカーシステム(2本1組)
AN-S100
標準価格 36,600円(税別)

システムラック

システムラック
(CZ-600C/601C/602C/603C/611C/612C/613C/623C用)
CZ-6SD1
標準価格 44,800円(税別)

※4 ご使用に際しては、あらかじめ別売の1MB増設RAMボードCZ-6BE1 標準価格35,000円(税別・CZ-600C用)、CZ-6BE1B 標準価格28,000円(税別・CZ-601C、CZ-611C、652C、653C、662C、663C用)を増設してください。
※5 モデムユニットCZ-8TM2に同梱のソフトはX1/X1ターボシリーズ用です。

品名	品番	価格
●ミニフロッピーディスクユニット(2D)	★CZ-502F	99,800円
●ミニフロッピーディスクユニット(2D・1ドライブ)	CZ-503F	49,800円
●増設用ミニフロッピーディスクドライブ(2D)※4	CZ-53F-BK	19,800円
拡張ボード・その他		
●モデムユニット(300/1200ボー)	CZ-8TM2	49,800円
●320KB外部メモリ	CZ-8BE2	29,800円
●RS-232C・マウスボード※5	CZ-8BM2	19,800円
●フロッピーディスクインターフェイス※6	CZ-8BF1	14,800円

品名	品番	価格
●JIS第1水準漢字ROM※7	CZ-8BK2	19,800円
●RS-232C用ケーブル(平行接続型)	CZ-8LM1	7,200円
●RS-232C用ケーブル(クロス接続型)	CZ-8LM2	7,200円
●拡張I/Oボックス	CZ-8EB3	33,800円
●RFコンバータ※8	AN-58C	2,980円
●インテリジェントコントローラ	CZ-8NJ2	23,800円
●マウス・トラックボール	CZ-8NM3	9,800円
●マウス	CZ-8NM2A	6,800円
●トラックボール	CZ-8NT1	13,800円

品名	品番	価格
●ジョイカード	CZ-8NJ1	1,700円
●チルトスタンド	CZ-6ST1-E・-B	5,800円
●高性能CRTフィルター※9	BF-68PRO	19,800円
●スキャナ用パラレルボード※10	CZ-8BN1	27,800円

●品番中の-表示は、B〈ブラック〉・E〈オフィスグレー〉を示します。※1 X1ターボZシリーズ用 ※2 CZ-862Cには接続できません。※3 X1ターボシリーズ用 ※4 CZ-830C用 ※5 X1シリーズ用 ※6 CZ-850CでCZ-520Fを使用する場合に必要 ※7 CZ-800C、801C、802C、803C、811C、820C用 ※8 CZ-820C、822C、830C用 ※9 14/15型用 ※10 CZ-8NS1用●接続等の説明につきましては、周辺機器総合カタログをご参照ください。

★印の商品は在庫僅少です。

子機器事業本部液晶映像システム事業部第2商品企画部〒162 東京都新宿区市谷八幡町8番地☎(03)260-1161(大代表)

SHARP

"アート"と呼べる高水準のソフトウェアが

次代のインテリジェンス、ウィンドウ環境をあなたのX68000で。

ユーザー本位の操作環境を提供するフル画面マルチウィンドウタイプの美しいデスクトップ(テキスト面/単色4階調+カラー4色、グラフィック面/カラー65,536色中16色)、イベント・ドリブン型マルチタスク処理により複数の作業を同時に処理できる疑似マルチタスクや入出力装置の設定が簡単に行える多機能コントロールパネルを搭載した本格ウィンドウシステムです。従来のビジュアルシェルとは異なり、今後のアプリケーションソフトが統一された操作環境で実行できるようになります。

SX-WINDOW ver1.0

CZ-259SS 10万台達成ご愛用感謝価格6,800円(税別)

高速通信をサポート。これからの、そしてさまざまな通信環境に対応する高機能コミュニケーションソフト。

Communication PRO-68Kのバージョンアップ版です。300BPSから19,200BPSまでの通信速度に対応し、パソコン同士の接続や各種データベースの漢字端末に、またホストコンピュータとのデータ通信に利用できます。さらにMNPモデムへの対応で、ハードフロー制御(CTS/RTS)をサポート。その他、高速逆スクロール機能、オートログイン/オートパイロットが可能な自動実行機能、コンカレント機能も装備。行入力機能やスクリーンエディタなど豊富な編集機能も魅力です。
また、バイナリファイルを転送するプロトコルとしてX modem(128/SUM、128/CRC、1K)、Ymodem(G、BATCH、G-BATCH)、TransIt2(TEXT、BINARY)プロトコルもサポートしています。

CZ-257CS

標準価格
19,800円(税別)

Communication PRO-68K ver2.0

ソースコードデバッガをはじめ、各種開発ツールを強化。バージョンアップされたCコンパイラ。

Cのソースレベルでデバッグできるソースコードデバッガを搭載したほか、各種開発ツールを強化した総合開発ツールです。また、ライブラリはHuman68k ver2.0の拡張DOSコールもサポートしているなど、よりX68000のハードウェアを活かせる豊富なライブラリ(約800種)となっています。強力なMAKEも新たに追加。C言語の標準であるANSI規格準拠をさらに強化し、プロトタイプ宣言もデフォルトに変更されました。「BASIC-Cコンバータ」、「アセンブラ」、「リンカ」、「デバッガ」、「ソースコードデバッガ」、「アーカイバ」、「ライブラリアン」、「コンバータ」などのツールが装備されています。

CZ-245LS

8月発売予定

C compiler PRO-68K ver2.0

シャープ株式会社 ●お問い合わせは…シャープ㈱電子機器事業本部液晶映像システム事業部第2商品企画部 〒162 東京都新宿区市谷八幡町8番地 ☎(03)260-1161(大代表)へ。

資料請求券 CZ-ソフト Oh!X 9係

X68000をサポート。

シャープオリジナルソフトウェア
X68000

ビジネスツール

Hyperword
■CZ-251BS　標準価格39,800円(税別)

X68000の優れたグラフィック環境を活用し効率的に文書を作成するためのインテリジェントワープロです。アイデアプロセッサ機能、ハイパーテキスト機能などをサポート。データの整理やプレゼンテーションツールなど幅広い用途に利用できます。

CYBERNOTE PRO-68K
■CZ-243BS　標準価格19,800円(税別)

プライベートなデータやビジネスデータを簡単な操作で管理・運営できるパーソナルデータベースです。リフィル、タックシール、ハガキなどへの印字もOK。シャープ電子手帳とのデータ交換可能(別売の通信ケーブルCE-200Lが必要)。

Stationery PRO-68K
■CZ-240BS　標準価格14,800円(税別)

他のソフトを起動する前に、このStationery PRO-68Kを一度起動するだけで、他のソフトを実行中にも「スケジュール」「住所録」など多彩な機能をワンタッチで使用できます。シャープ電子手帳とのデータ送受信も実現。(別売の通信ケーブルCE-200Lが必要)。

TOP給与計算エキスパート
■CZ-228BS標準価格200,000円(税別)

給与計算から明細発行までを、リアルイメージ入力により自動的に、素早く処理することができます。

CARD PRO-68K
■CZ-226BS　標準価格29,800円(税別)

自由なレイアウト画面で入力できるワープロ機能を装備したカード型リレーショナルデータベース。

DATA PRO-68K
■CZ-220BS　標準価格58,000円(税別)

入力の手間を軽減するヒストリー機能を装備した、コマンド型リレーショナルデータベースです。

TOP財務会計
■CZ-227BS標準価格200,000円(税別)

会計エキスパートシステムとデータベースを搭載し、機能と操作性を両立させた財務会計ソフト。

CARD PRO-68K用システム手帳リフィル集
■CZ-241BS　標準価格9,800円(税別)

CARD PRO-68K用活用フォーム集
■CZ-242BS　標準価格9,800円(税別)

BUSINESS PRO-68K
■CZ-212BS　標準価格68,000円(税別)

スプレッドシート(表計算)、データベース、グラフ作成機能を一体化させた統合ビジネスツールです。

アートツール

NEW PrintShop PRO-68K
■CZ-221HS　標準価格19,800円(税別)

オリジナリティあふれるはがき等、簡単に作成、印刷できるホームプロダクティビリティツール。ほとんどの処理をアイコンで表示しマウスで選ぶフレンドリーオペレーション。

グラフィックライブラリ VOL.1
■CZ-235GS　標準価格8,800円(税別)

暑中見舞用を中心としたNEW Print Shop PRO-68K用グラフィックデータ集。

グラフィックライブラリ VOL.2
■CZ-236GS　標準価格8,800円(税別)

年賀状を中心としたNEW Print Shop PRO-68K用グラフィックデータ集。

サウンドツール

Musicstudio PRO-68K ver.1.1
■CZ-252MS　標準価格28,800円(税別)

24トラック対応MIDIマルチレコーディングソフトMusicstudio PRO-68Kがバージョンアップしました。従来の機能に加え、小節間のコピー及びデリートや、MIDIインプットモニターなど、数々の機能を追加・改良。さらに使いやすくなりました。

※MIDIボード(CZ-6BM1)が必要です。

MUSIC PRO-68K〔MIDI〕
■CZ-247MS　標準価格28,800円(税別)

MIDI対応自動伴奏機能をサポート、簡単な楽譜入力でMIDI演奏が楽しめます。

※MIDIボード(CZ-6BM1)が必要です。

ソングライブラリ〈101曲集〉
■CZ-248MS　標準価格8,800円(税別)

鑑賞用と音楽データ加工作成用からなるライブラリです。

Sampling PRO-68K
■CZ-215MS　標準価格17,800円(税別)

AD PCM機能を活かす高機能サンプリングエディタ。多彩なEDITORを装備、サンプリング音のデータはBASICでも活用できます。

SOUND PRO-68K
■CZ-214MS　標準価格15,800円(税別)

スタジオのコンソールパネルを操作する感覚でFM音源による音創りが楽しめるサウンドエディタ。

MUSIC PRO-68K
■CZ-213MS　標準価格18,800円(税別)

最大8パートのスコア(総譜)が書け、内蔵のFM音源で演奏できる楽譜ワープロ＆演奏用ツール。

ゲーム

シューティングゲーム
〈ツインビー〉
■CZ-217AS
標準価格7,800円(税別)
©KONAMI 1988

シューティングゲーム
〈沙羅曼蛇〉
■CZ-218AS
標準価格8,800円(税別)
©KONAMI 1989

ブロックゲーム
〈アルカノイド〉
■CZ-222AS
標準価格7,800円(税別)
©TAITO CORP. 1987

ドライブゲーム
〈フルスロットル〉
■CZ-231AS
標準価格8,800円(税別)
©TAITO CORP. 1988

スポーツゲーム
〈熱血高校ドッジボール部〉
■CZ-232AS
標準価格7,800円(税別)
©TECHNOS JAPAN CORP. 1988

アクションゲーム
〈パックマニア〉
■CZ-233AS
標準価格7,800円(税別)
©NAMCO

アクションゲーム
〈ニュージーランドストーリー〉
■CZ-230AS
標準価格8,800円(税別)
©TAITO CORP. 1989

スポーツゲーム
〈V'BALL〉
■CZ-246AS
標準価格7,900円(税別)
©TECHNOS JAPAN CORP. 1989

バイクレーシングゲーム
〈スーパーハングオン〉
■CZ-238AS
標準価格8,800円(税別)
©SEGA 1987

ジェットヘリ・シミュレーションゲーム
〈サンダーブレード〉
■CZ-239AS
標準価格9,500円(税別)
©SEGA 1987

アクションゲーム
〈ダウンタウン熱血物語〉
■CZ-254AS
標準価格8,800円(税別)
© TECHNOS JAPAN CORP. 1989

開発ツール

OS-9/X68000
■CZ-219SS　標準価格29,800円(税別)

OS-9のもつマルチタスク機能、リアルタイム機能を活かした使い易く機能的なOS環境を提供。これまでのデータ資産も活かせます。

※OS-9はマイクロウェア社の登録商標です。

Human68k ver2.0
■CZ-244SS　標準価格9,800円(税別)

THE福袋V2.0
■CZ-224LS　標準価格9,980円(税別)

AI-68K(Staff LISP/OPS PRO-68K)
■CZ-234LS標準価格188,000円(税別)

BG EDITモード

「G=ツール」　。

新発売

背景

ZAINSOFT

資料 オートデモ 請求券 OhX 9月号

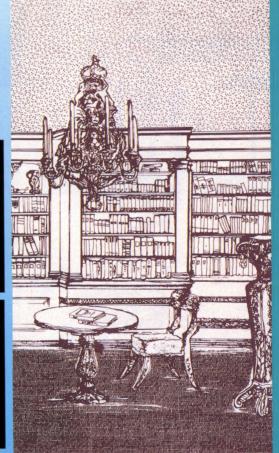
本格的ファイルマネージングソフトウェア

業界の新星、ロゴスシステムが
ユーザーの希望を1つの形にしました。
これは必要だとか便利じゃない、快感だ！

全国有名パソコンショップでお求め下さい。
電話１本での通信販売も受付いたしております。

THE FILE PROFESSORの実力

ディスクのバックアップ、ディスクのエディット、ディスクの初期化、ディスクの比較、ディスクの検査、ディスクの情報、ＦＡＴのエディット、ファイルの検索、ディレクトリのコピー、ディレクトリの削除、ヴォリュームラベルの設定、ディレクトリの作成、ディレクトリ構造の再読み込み、ディレクトリ構造の印刷、ディレクトリ名の変更、ディレクトリ内容のソート、削除ファイルの復元、ファイル属性の変更、ファイルのコピー/移動、ファイルの削除、ファイルのエディット、ファイルの配置情報、ファイル一覧の印刷、ファイル名の変更、ファイルのソート、ファイル更新日時の変更、ファイルの表示、ファイルの奨行、カレンダー、ハードディスクの直接エディット、システム情報の表示、コマンドシェル、現在時刻の変更。

メニュー選択方式を実現!!
初心者でも簡単に使える
（画面写真は、98用を開発中のものです。）

ロゴスシステム

このソフトはロゴスシステムのデビュー作です。でも、だからといってなめてもらっちゃぁ困ります。私達は、いろいろなソフトを作りました。そのどれもが他社から発売されていました。出来る事ならば自分達で発売したい！その願いがやっとかないました。

The File Professor

好評発売中！

ロゴスシステム
〒615 京都市右京区西院上今田町17-1 L&Pビル4F
TEL (075) 812-6383　FAX (075) 822-6915

定価28,000円

闇の血族

THE PREDESTINED HOMICIDES #1

美少女名探偵 魅由の繰り広げる
ミステリアスアニメーションアドベンチャー第1弾!!

艶やかなファッション界を襲う奇怪な連続殺人事件。
南米の血に隠された秘密とは?
そして魅由を待ち受ける血族の宿命は?

上巻

あたし、魅由。
新宿にあるデザイン・スタジオの、新人A・D(アパレル・デザイナー)。……なんだけどあたしの持ってる妙な「力」みたいなモノ——人の心が判っちゃったり、変にカンが良かったり——のせいで、周りからは「名探偵魅由」なんて呼ばれて、よく相談事を持ち込まれたりしている。で、そんなある日、友達のモデルが、突然、殺されてしまった。
そして、あたしの親友だった唯も……!
これって……ひょっとして連続殺人事件ってヤツ?!

新発売!!

X68000対応 5"-2HD
●ローランド社MT-32・CM-64完全対応
MIDIインターフェイスボードCZ-6BMI
又は、SACOM製SX-68Mが必要です。
(初期のMT-32では、正常に演奏できません。)
標準価格 **8,800**円

システムサコムでは、ゲームソフトのクリエーターを募集しています!
ゲーム造りの喜び哀しみを一緒に味わってみませんか!

●プログラマー (8086や68000のアセンブラやC言語が使える人) ●グラフィック (原画やドット絵を描ける人) ●ビジュアリスト(オープニングデモなどの構成が出来る人)
●ゲームデザイナー (ゲームの企画書を作成出来る人)
応募方法 履歴書と供に作品等(プログラムやドット絵、シナリオ等)を当社にお送り下さい。書類選考の上面接日を御連絡致します。 資格 高校卒以上30才程までの方。 給与 能力や経験により優遇致します。 待遇 昇給年1回、賞与年2回、通勤手当、残業手当、各種保険完備。 勤務時間 午前9時～午後5時30分。 休日 完全週休2日制、年末年始、夏季休暇、年次有給休暇有。

38万キロの虚空CD
東芝EMIより
新発売!!
●MT 税込価格2,250円
●CD 税込価格2,530円

ジェミニウイング X68000対応 バリバリ開発中!

ノヴェルウェアシリーズ		
DOME	PC-8801SR	各 9,800円
	PC-9801	
	X-1	
	X68000	
	FM-7・77AV	
	MSX2	
CHATTY	PC-8801SR	8,800円
ソフトでハードな物語	PC-9801	7,800円
	X-68000	7,800円
	FM-TOWNS	9,800円
ソフトでハードな物語2	PC-9801	7,800円
	X-68000	7,800円
	FM-TOWNS	9,800円
38万キロの虚空	PC-9801	各 9,800円
	X-68000	
	FM-TOWNS	

好 評 発 売 中!!		
ユーフォリー	X-1	6,800円
エボリューション	FM-TOWNS	9,800円
幽 霊 君	MSX2	6,800円
プロヴィデンス	PC-8801SR	7,800円
ヴ ァ ル ナ	PC-8801SR	7,800円
メタルサイト	X-68000	8,800円
史上最強のビデオバイブル	FM-TOWNS	4,800円

※標準価格には消費税は含まれておりません。

株式会社 システム サコム
〒130 東京都墨田区両国4-38-16
両国桜井ビル4F
ハードウェア部 03(635)5145
ソフトウェア部 03(635)7609

X68000の威力。「幻獣鬼」が今発揮する。

UNDEADLINE

幻獣鬼™

げんじゅうき

X68000

X68000

ACT-neXt………幻獣鬼

鋭い！ X68000ユーザーの鋭き感性をより研ぎ澄ます
鋭い！ 魂より出ずる鋭き野望が渦巻く世界
鋭い！ プレイヤーの鋭きテクニックがすべての明暗を分かつ

9/14 FRI 新発売

●X68000 5"2HD 3枚組

標準価格 ¥7,800

＊表示価格に消費税は含みません

＊写真は開発中のもので

X68000の本質。
「黒衣の貴公子」が今解き明かす。

黒衣の貴公子

X68000
RPG-neXt
ルーンワース 黒衣の貴公子

好評発売中

熱い！ X68000ユーザーの熱き要望に応え堂々登場!!
熱い！ ルーンワースとよばれる異世界で繰り広げられる熱き冒険譚
熱い！ プレイヤーの熱き魂が物語を自由に織りなしてゆく

●X68000 5"2HD 3枚組
- ●全グラフィック描き起こし(高解像グラフィック512×512ドット)
- ●ジョイスティック対応
- ●FM音源8音＋ADPCM音源対応

●PC-9801VM、UVシリーズ PC-286、386シリーズ、NOTE対応
5"2HD/3.5"2HD 2枚組 ●サウンドボード対応●ジョイスティック対応
※要640Kバイト
※PC-9801/E/F/M/VF/U2/XA、PC-286L/LE/LFおよびPC-286NOTE Executiveでは、ドライブ、RAM等の増設の如何にかかわらず、作動いたしません。

●PC-8801SRシリーズ・VA、98DO対応 5"2D 5枚組

● MSX2/MSX2+ (RAM64K以上、VRAM128K以上) 3.5"2DD 3枚組

標準価格 各¥8,800
※表示価格に消費税は含みません。

RPG-neXt………ルーンワース 黒衣の貴公子
ACT-neXt………幻獣鬼
SLG-neXt………遙かなるオーガスタ

■通信販売ご希望の方は現金書留で料金と商品名・機種名と電話番号を明記の上、当社迄お送りください。(速達希望の方は300円プラス)
■カタログご希望の方は、送料として切手200円分を同封の上、カタログ請求券をお送りください。(葉書での請求はお断わりします)
●T&Eの最新情報がわかるテレフォンサービス 名古屋(052)776-8500

T&E SOFT
企画・開発・製造・販売
株式会社 ティーアンドイーソフト
〒465 名古屋市名東区豊が丘1810番地 PHONE:052-773-7770

RPGの概念を一変させた傑作!

1988年発売と同時に世界中のゲーム・フリークを熱狂させた、あの「ダンジョン・マスター」が今、日本中を荒しまわる。
3Dグラフィックスによる複雑な迷路、数々の謎、パーティーを突然襲って来るモンスター。
そしてなによりもプレイヤーの考えること、見ること、手にすること、
すべてにリアルタイムで動いていく……本当の意味のリアルRPGだ。

なぜ、世界をそして日本をこれ程までに興奮させたのか!
その答えは君自身で出して欲しい。

もう逃げられない!

Dungeon ダンジョン・マスター Master

※画面写真はX-68000版

好評発売中 ■X68000 マウス対応 ■PC-9801VM21/11, VX, RX, RS, RA ■PC-98DO ■PC-9801UV21/11, UX, CV, EX, ES 要バス・マウス/アナログRGB対応 各¥9,800(税抜)

★「ダンジョン・マスター」の本8月中旬発売!
❶「ダンジョン・マスター ガイドブック」発行:秀和システムトレーディング㈱ 予価:¥1500 ❷「ダンジョン・マスター パーフェクトガイド」発行:ナツメ出版企画㈱ 予価:¥1800

これが進化した麻雀ソフト、待望のX-68対応発売。

雀豪2 強知能版

麻雀ソフトの決定版登場!プレイすればするほど個性をもったプレイヤーに成長する自己成長型サンプリング機能と、より強化された推論型人工知能の搭載で限りなく実戦麻雀に近づいた。
リアルな4人囲みと見やすい麻雀牌、迫力ある効果音などの採用が麻雀ソフトの金字塔の地位を不動のものにする。

■8月24日発売:X-68000 ■好評発売中:PC-9801シリーズ
各¥9,800(税抜き)

※画面写真はX-68000版の開発画面です

●発売 ビクター音楽産業株式会社

通信販売 当社の商品をお近くのパソコンショップでお買い求めになれない場合、商品名、機種名、住所、氏名、電話番号を明記のうえ、下記住所へ定価プラス3%消費税分を現金書留にてお申し込み下さい。(送料無料) 〒151 東京都渋谷区千駄ヶ谷2-8-16 ビクター音楽産業㈱〈通信販売〉

掲載商品2万円以上送料無料!!

ツクモは先取り!

ツクモ通販受注センターフリーダイヤル **0120(377)999**
商品のお問い合せは各店又は通販部☎03(251)9911へ

冬のボーナス一括払・金利手数料無!受付中!!詳しくはお問い合せ下さい。

毎年恒例のツクモの日特価品!

9/1(土)▼9/9(日)

アナログジョイスティック	48ドットカラー熱転写プリンター(ケーブル付)	40MBハードディスク
定価¥23,800	定価¥99,800	定価¥102,000
ツクモの日特価 ¥19,900	ツクモの日特価 ¥69,800	ツクモの日特価 ¥59,800

「MIDIサウンドライブ in オータム」
♬大好評のMIDIライブショーの第2弾デス!

開催日 9月22日(土)・23日(日) 午後3回(12:30/1:30/2:30)30分づつ。
場所 九十九電機(株)7号店1階店頭にて、X68000とMIDI楽器を使った楽しい演奏♬クイズもあって景品が当る!是非お立ち寄り下さい。

All in Note
予約受付中!!

AX286N-H2
定価¥398,000
- 「Business Mate」標準装備
- 20MバイトHD搭載
- フリートップサイズ
- 小さいボディに高性能

周辺機器
3.5インチフロッピーディスクドライブ UE-1F04 定価¥49,800
一体型外部バッテリ UE-1X07 定価¥26,000
表計算ソフト
Microsoft EXCEL Ver.2.1 定価¥98,000
ワープロソフト
一太郎AX 定価¥68,000
書院AX(UE-6Z10) 定価¥49,800
★7号店で取扱っています。

X68000シリーズ ツクモ特価販売中!!

PROII CZ653C 定価¥285,000 / CZ663C 定価¥395,000
●次世代のインテリジェンス、SX-WINDOW搭載 ●知的ニュースタンダードフォルム ●BIOSの改良によりハイスピード処理を実現 ●2Mバイトの大容量メモリを標準装備 ●拡張I/Oポート4スロット標準装備

EXPERTII CZ603C 定価¥338,000 / CZ613C 定価¥448,000
●次世代のインテリジェンス、SX-WINDOW搭載 ●象徴のフォルム、マンハッタンシェイプ ●BIOSの改良によりハイスピード処理を実現 ●3Mバイトの大容量メモリを標準装備

SUPER HD CZ623C 定価¥498,000
●次世代のインテリジェンス、SX-WINDOW搭載 ●「チタン」カラーのクォリティブラック ●80MB SCSIハードディスク搭載 ●世界標準SCSIインターフェース標準装備 ●BIOSの改良によりハイスピード処理を実現 ●3Mバイトの大容量メモリを標準装備

X68000用メモリーボード

1MB増設用メモリーボード ACE & PROシリーズ内蔵用
ツクモ特価 ¥19,800 (消費税別途¥594)

2MB増設RAMボード……ツクモ特価¥42,500 (消費税別途¥1,275)
4MB増設RAMボード……ツクモ特価¥72,500 (消費税別途¥2,175)
※2MBと4MBは全シリーズ対応。拡張スロット用。

ハードディスク

シャープ
光磁気ディスクユニット
CZ-6MO1 予約受付中!
SCSIボード
CZ-6BS1 予約受付中!

アイテック (カラー:ブラック/グレー)
IT X640 定価¥158,000 特価¥89,800
IT X680 定価¥198,000 特価¥118,000

♪♪♪ X68000用MIDI ♬♬♬

Aセット		Bセット	
CM-32L	¥69,000	CM-64	¥129,000
SX-68M	¥19,800	SX-68M	¥19,800
Musicstudio Mu-1	¥19,800	Musicstudio Mu-1	¥19,800
合計定価	¥108,600	合計定価	¥168,600
ツクモ特価	¥91,800 (消費税別途¥2,754)	ツクモ特価	¥144,000 (消費税別途¥4,320)
クレジット例(税込)	月々¥5,830×18回払	クレジット例(税込)	月々¥7,107×24回払

★Musicstudio PRO-68K V1.1又は、Music PRO68K(MIDI)のソフトの場合には¥8,000プラスになります。

おすすめソフト & 新作ソフト

Hyper WORD 定価¥39,800
SX-WINDOW 定価¥6,800
CCompilerPRO-68K Ver 2.0 予約受付中!
Simcity (9/7発売予定) 予約受付中!
サイバリオン (シャープ シューティングゲーム) 予約受付中!

Software tools

GRAPHIC TOOLS
マジックパレット 特価¥16,830
Z's STAFF PRO-68K 特価¥49,300
サイクロンExpressα68 特価¥83,300
デジタルクラフト 特価¥33,800

電子手帳ソフト
CYBERNOTE PRO-68K 特価¥16,830
Stationery PRO-68K 特価¥12,580
※通信ケーブル CE-300L 特価¥2,520

通信モデム & ソフト

45% off
アイワ
PV-A24MNP5 限定特価¥29,800
定価¥54,800 (消費税別途¥894)
た〜みのる2 ツクモ特価¥15,000
定価¥17,800 (消費税別途¥450)

電子手帳 & ポケコン

PA-8600 特価¥24,800
PA-7500 特価¥17,800
PA-6500GY 特価¥9,800
新製品 PC-E550 特価¥28,800

大好評入会者募集!
ツクモグローバルカード

18才以上なら学生でもOK!
国内・外で活躍!
使って便利、持ってて安心!ツクモグローバルカードはジャックス・VISA、セントラル・マスターとの提携カードです。ツクモ各店でのお買物がらくらくできるうえに、国内はもとより海外でのショッピングもOK!しかも18歳以上なら学生でもOK!
冬のボーナス一括払・金利手数料無!受付中!!
お申し込みは(03)251-9898又は各店で

秋葉原各店
営AM10:15〜PM7:00 休毎週木曜日

ツクモは「スーパーX PRO SHOP」です。
PRO STAFF ツクモ
九十九電機(株)
〒101-91 東京都千代田区神田郵便局私書箱135号

7号店 荒井 / N.C店 福地

ツクモ7号店 ☎03-253-4199(担当/荒井)
便利で安心な通信販売
通信販売部☎03-251-9911
■ニューセンター店 ☎03-251-0987(担当/福地)
■ツクモ5号店 ☎03-251-0531(担当/川名)
■名古屋1号店 ☎052-263-1655(担当/吉高)
■名古屋2号店 ☎052-251-3399(担当/横山)
■ツクモ札幌 ☎011-241-2299(担当/村井)

★表示価格には消費税は含まれておりません。
★商品のご注文は在庫確認の上お願いします。

カード払い	全国代金引き換え配達	クレジット払い	現金書留払い	銀行振込払い	各種リース払い
通信販売での御利用カード、ツクモグローバルカード、VIPカード、セントラル、ジャックス※御本人様より電話で通信販売部へお申し込み下さい。	お申し込みは☎03-251-9911へお電話1本!配達日の指定もできます。	月々¥3,000以上の均等払いも頭金なし、夏・冬ボーナス2回払いも受付中!	〒101-91 東京都千代田区神田郵便局私書箱135号 九十九電機(株)通信販売部 oh!X係	事前に☎でお届け先をご連絡下さい。富士銀行 神田支店(普)No.894047 九十九電機(株)	くわしくは各店にお問い合せ下さい。ケースに合わせてご相談にのらせて頂きます。

■店頭にて、ゲームソフト25%OFF!!(税別)、超低金利 オクトハッピークレジットをご利用下さい!!

■朗報です。冬のボーナス一括払い(12月末)OK!!手数料なし。(翌月末払いもOK!!)ご利用下さい。

パソコンプラザ

OCT オクト

案内図

至品川 東口ロータリー JR蒲田から徒歩5分 京浜蒲田から徒歩1分 至品川

三菱BK 商店街アーケート 2F JR蒲田駅 京浜蒲田駅 富士BK 三和BK 日本信託 八幡神社 環八通り NTT 至横浜 至横浜

店頭セール実施中

'90オクトで始まるパソコンワールド

☎03-730-6271

●営業時間 AM 11:00〜9:00/日曜・祭日PM7:00　電話一本で、ハイ即納

〒144 東京都大田区蒲田4-6-7 FAX 03-730-6273

●定休日毎週火曜日 祭日の場合翌日になります。

全国通販

オクト ラクラククレジット

1回	2%	3回	2.5%	6回	3.5%	10回	5%	12回	5%	15回	7.5%
18回	9%	20回	10%	24回	11%	30回	14.5%	36回	15.5%	48回	20%

OCT-1 システム インフォメーション

- ▶全商品保証付(メーカー保証)
- ▶超低金利ハッピークレジット(1回〜60回)頭金ナシOK!
- ▶ボーナス一括払いOK!ボーナス2回払いOK!!
- ▶配達日の指定OK!(万全なサポート体制)
- ▶商品の組合せ自由!オクトフリーダムシステム
- ▶店頭デモンストレーション実施中

オクト セレクテッドシステム

広告掲載商品以外の製品も取扱っております。

OCT-1 オクト-1 蒲田 OPEN

●冬のボーナス一括払い(12月末)OK!!
手数料なしです。絶対、お得ですゾ。
翌月末払いも受付けています(9月末)

★下記セットでお買い上げの方にはプレゼント!! ●①MD-2HD 10枚②ジョイカード 2個(連射式)③シリコンキーボードカバー

お好みのセットをお選び下さい。

送料無料

●SX-WINDOW搭載。
●40Mバイトハードディスク搭載

EXPERTII・EXPERTII-HD

●CZ-603C-BK/GY
定価¥338,000
●CZ-613C-BK/GY
定価¥448,000

現金特価!! 推選
お電話下さい。

●SX-WINDOW搭載。
●拡張I/Oポート4スロット装備

PROII・PROII-HD

●CZ-653C-BK/GY
定価¥285,000
●CZ-663C-BK/GY
定価¥395,000

CZ-8NJ2
●インテリジェントコントローラ
定価¥23,800
超特価¥16,800

15型カラーディスプレイTV
CZ-605D-GY/BK
定価¥115,000

15型カラーディスプレイTV
CZ-613D-GY/BK
定価¥135,000

14型カラーディスプレー
CZ-604D-GY/BK
定価¥94,8000

21型カラーディスプレイ
CU-21HD
定価¥148,000

Ⓐ CZ-603C+CZ-605D…………定価合計¥453,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓑ CZ-613C+CZ-605D…………定価合計¥563,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓒ CZ-653C+CZ-605D…………定価合計¥400,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓓ CZ-663C+CZ-605D…………定価合計¥510,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓔ CZ-603C+CZ-613D…………定価合計¥473,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓕ CZ-613C+CZ-613D…………定価合計¥583,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓖ CZ-653C+CZ-613D…………定価合計¥420,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓗ CZ-663C+CZ-613D…………定価合計¥530,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓘ CZ-603C+CZ-604D…………定価合計¥429,800▶オクト大特価

12回	¥28,000	24回	¥14,800	36回	¥10,200	48回	¥8,000

Ⓙ CZ-613C+CZ-604D…………定価合計¥542,000▶オクト大特価

12回	¥36,000	24回	¥19,000	36回	¥13,100	48回	¥10,200

Ⓚ CZ-653C+CZ-604D…………定価合計¥379,800▶オクト大特価

12回	¥25,400	24回	¥13,400	36回	¥9,300	48回	¥7,200

Ⓛ CZ-663C+CZ-604D…………定価合計¥489,800▶オクト大特価

12回	¥32,200	24回	¥17,000	36回	¥11,800	48回	¥9,200

Ⓜ CZ-603C+CU-21HD…………定価合計¥486,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓝ CZ-613C+CU-21HD…………定価合計¥596,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓞ CZ-653C+CU-21HD…………定価合計¥433,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

Ⓟ CZ-663C+CU-21HD…………定価合計¥543,000▶オクト大特価

12回	?	24回	?	36回	?	48回	?

♡どんどんTELしよう。安くなるかもヨ!!

♡クレジット価格は、消費税込みですヨ。ご利用下さい!!

※クレジットの回数は1回〜60回、ボーナス併用などありますのでお電話でお問合せ下さい。

■本体セット:送料無料　●店頭デモ実施中…専門の係員が詳細にアドバイス致します。ぜひご来店下さい。

※上記料金には、消費税は含まれておりません。消費税が付加されますので、詳しくは、電話でお問合せ下さい。

■便利です。夜9時まで営業しております。お立寄り下さい。お待ちしております!!

■店頭にて、ゲームソフト25%OFF!!（税別）、超低金利 ハッピークレジットをご利用ください!!
■特に人気のある商品によっては、しばらくお待ち願うことがありますのでご了承下さい。

厳選された製品を、より安く、より早く、皆様のお手元に!!

広告掲載商品以外の製品も取扱っております。

チャンス！X68000・SUPER-HD（チタン）＝好評・発売中 どんどんTEL下さいネ。 送料ナシ!!

SX-WINDOW搭載。

●ザ・ワークステーションと呼ぶにふさわしいスーパーな68000.!! 新登場.!! SUPER-HD。

※プレゼント！①MD-2HD10枚 ②アフターバーナー（￥9,200） ③ジョイカード（連射式） ④シリコンキーボード（￥2,800）

X68000 SUPER-HD

●CZ-623C-TN＋CZ-613D-TN
定価合計￥633,000…大特価.!! TEL下さい。

※マウス・トラックボール付.!! ディスプレイにはスピーカ2個、チルト台付.!!

他のディスプレイ① CZ-602D、② 612D、③ CZ-603D、④ CU-21HDの組合せもございますのでお問い合せ下さい。

12回	?	24回	?	36回	?	48回	?

♡安くてゴメンなさい。今だけヨ.!!

※超低金利クレジットご利用下さい。1回～60回払い、頭金ナシ！ボーナス1回払い、ボーナス2回払いOK！

X68000 EXPERT-HD

オクト限定スペシャルセット

ラストチャンス!! 早い者勝ち!!

●CZ-612C（BK）（￥466,000）
●CZ-602D（BK）（￥99,800）
●MD-2HD 10枚
●ジョイカード（連射式×2個）
●シリコンキーボード・カバー

オクト超特価
￥364,000（送料・消費税込み.!!）

※ディスプレイ＝①CZ-604D ②CZ-605D ③CZ-613D ④CU-21HD との組合せもございます。TEL下さい。

オクト特選　シャープ周辺機器　（送料￥1,000）

●CZ-6BE1 1MB増設RAMボード……（￥35,000）▶特価￥26,500
●CZ-6BE1B 1MB増設RAMボード………（￥28,000）▶特価￥21,000
●CZ-6BE2 2MB増設RAMボード……（￥79,800）▶特価￥60,500
●CZ-6BE4 4MB増設RAMボード……（￥138,000）▶特価￥104,800
●CZ-6BF1 増設用RS-232Cボード……（￥49,800）▶特価￥38,500
●CZ-6BG1 GP-IBボード……（￥59,800）▶特価￥45,000
●CZ-6BM1 MIDIボード……（￥26,800）▶特価￥20,500
●CZ-6BN1 スキャナ用パラレルボード‥（￥29,800）▶特価￥22,800
●CZ-6BP1 数値演算プロセッサボード（￥79,800）▶特価￥60,500
●CZ-6BO1 ユニバーサルI/Oボード…（￥39,800）▶特価￥30,500
●CZ-6EB1/BK 拡張I/Oボックス……（￥88,000）▶特価￥66,800
●CZ-6VT1/BK カラーイメージ・ユニット…（￥69,800）▶特価￥53,000
●CZ-6BL2 LANボード……（￥298,000）▶大特価
●CZ-8NM2A マウス……（￥68,800）▶特価￥5,300
●CZ-8NT1 マウストラックボール‥（￥98,800）▶特価￥7,500
●CZ-8NS1 カラーイメージスキャナ……（￥188,000）▶大特価
●CZ-6BC1 FAXボード……（￥79,800）▶特価￥60,500
●CZ-8TM2 モデムユニット……（￥49,800）▶特価￥38,000
●CZ-64H 増設ハードディスク…（￥120,000）▶大特価
●CZ-6TU GY/BK RGBシステムチューナー…（￥33,100）▶特価￥25,000
●BF-68PRO 高性能CRTフィルター……（￥19,800）▶特価￥15,500
●SX-68M（システムサコム） MIDIボード……（￥19,800）▶特価￥15,000
●PIO-68BE1-A（I/O DATA） 1MB増設RAMボード……（￥25,000）▶特価￥18,500
●PIO-6BE2-2M（I/O DATA） 2MB増設RAMボード……（￥50,000）▶特価￥37,000
●PIO-6BE4-4M（I/O DATA） 3MB増設RAMボード……（￥88,000）▶特価￥65,000
●CZ-6BV1 ビデオボード……（￥21,000）▶特価￥15,800

オクト面白グッズ

アイテック（送料￥1,000）
●IT-X640（￥158,000）……特価￥103,000
●IT-X680（￥198,000）……特価￥134,000

モデムコーナー（送料￥1,000）

●MD-1200AIII……特価￥14,800
●MD-24FS4……特価￥31,500
●MD-24FS5……特価￥34,800
●MD-24FP4……特価￥27,900
●MD-12FS……特価￥15,000

熱転写カラー漢字プリンター（ケーブル用紙付） 送料￥1,000

CZ-8PC4 ￥99,800 限定
●48ドット
サーマルヘッド
●B5～B4まで
●ハガキ可能
●カラー対応

オクト大特価 ~~￥64,800~~

①CZ-8PK10（24ピン漢字プリンター136桁）
定価￥97,800……大特価.!! TEL下さい。
②CZ-8PG1（24ピンカラー漢字プリンター80桁）
定価￥130,000……大特価.!! TEL下さい。
③CZ-8PG2（24ピンカラー漢字プリンター136桁）
定価￥160,000……大特価.!! TEL下さい。
④IO-735×（カラーイメージジェット）
定価￥248,000……大特価.!! TEL下さい。

パソコンラック 推奨 送料無料

①五段キャスター付

5段キャスター付
キーボードが収納できるから、手元でマウス操作がラクラクできる
棚板5段のマルチに活用できるディスク。
ウーン、こいつはデキル！
1325（H）×640（W）×700（D）
特価￥16,000

②四段キャスター付

4段キャスター付
どんなパソコンにもフレキシブルに対応！
使い易いデスクです。
1245（H）×614（W）×600（D）
特価￥12,000

③三段キャスター付

3段キャスター付
場所を選ばない簡易で便利なディスクです。
限定
1175（H）×640（W）×600（D）
特価￥8,800

X68000ソフト大セール実施中※ゲームソフトオール25%off

〈グラフィック〉●Z's STAFF PRO68K Ver.2.0（シャフト）定価￥58,000
オクト特価￥40,000
〈データーベース〉●KAMIKAZE（サムシンググッド）定価￥68,000
オクト特価￥46,000
〈グラフィック〉●C-TRACE68（キャスト）定価￥68,000
オクト特価￥51,000
〈C言語〉●C & Professional Pack（マイクロウェアジャパン）定価￥58,000
オクト特価￥44,000
〈グラフィック〉●サイクロン エキスプレス 定価￥78,000
オクト特価￥58,000
〈グラフィック〉●デジタルクラフト 定価￥39,800
オクト特価￥28,000
〈ワープロ〉●ハイパーワード 定価￥39,800 CZ-251BS
オクト特価￥29,800

型名	商品	定価	特価
CZ-211LS	Ccompiler PRO-68K	￥39,800	￥28,800
CZ-212BS	BUSINESS PRO-68K	￥68,000	￥48,000
CZ-213MS	MUSIC PRO68K	￥18,800	￥13,500
CZ-214MS	SOUND PRO-68K	￥15,800	￥11,500
CZ-215MS	Sampling PRO-68K	￥17,800	￥12,800
CZ-219SS	OS-9/X68000	￥29,800	￥21,000
CZ-220BS	DATA PRO-68K	￥58,000	￥41,000
CZ-257CS	Print Shop PRO68K.V.2	￥19,800	￥14,300
CZ-223CS	Communication PRO-68K	￥19,800	￥14,300
CZ-224LS	THE 福袋 V2.0	￥9,900	￥7,500
CZ-226BS	CARD PRO-68K	￥29,800	￥21,300
CZ-241BS	システム手帳リフィル集	￥9,800	￥7,500
CZ-242BS	活用フォーム集	￥9,800	￥7,500
CZ-244SS	Homan 68K Ver.2.0	￥9,800	￥7,500
CZ-247MS	MUSIC PRO-68K〔MIDI〕	￥28,800	￥20,800
CZ-240BS	Stationery PRO-68K	￥14,800	￥11,500
CZ-243BS	CYBER NOTE PRO-68K	￥19,800	￥15,200
EW		￥38,000	￥29,800
G-68K		￥14,800	￥11,400
E-68		￥19,800	￥15,300

★オクト今月だけの新品限定販売（各1台限）（送料￥1,000）

● CZ-611C（BK）定価￥399,800……大特価￥218,000
● CZ-652C（BK）定価￥298,000……大特価￥188,000
● CZ-662C（BK）定価￥408,000……大特価￥248,000
● CZ-601D（BK）定価￥119,800……大特価￥68,000
● CZ-601D（GY）定価￥119,800……大特価￥68,000
● CZ-612D（GY）定価￥119,800……大特価￥74,000
● IO-735 定価￥248,000……大特価￥158,000

店頭ゲームソフトオール25%off！ビジネスソフト 25%より特価中

●尚、送料として1ケ￥500、2ケ￥700、3ケ以上で￥1,000となります。（税別）

★通信販売お申込みのご案内★ 〒144 東京都大田区蒲田4-6-7 TEL：03-730-6271

お申込みはお電話でお願いします。お客様の〈住所〉〈氏名〉〈電話番号〉及び〈商品名〉をお知らせ下さい。●入金確認後ただちに商品をご送付いたします。

現金一括払い
銀行振込：お近くの銀行より（電信扱い）にてお振込み下さい。
現金書留：封筒の中に住所・氏名・商品名をご記入の上当社までお送り下さい。

クレジット
専用お申込用紙をお送り致します。ので、必要事項をご記入、ご捺印の上ご返送下さい。手続きは簡単です。

オクト ラクラク クレジット表

1回	2%	3回	2.5%	6回	3.5%	10回	5%
12回	5%	15回	7.5%	18回	9%	20回	10%
24回	11%	30回	14.5%	36回	15.5%	48回	20%

振込先
富士銀行 久ヶ原支店 ㊟No.1824
三菱銀行 蒲田支店 ㊟No.0278691
株式会社 億人（オクト）

※掲載の価格は変動しますので、まずは、お電話にてご確認ください。
※連休のお知らせ＝8/21（火）、22（水）は連休です。
※上記料金には、消費税は含まれておりません。消費税が付加されますので、詳しくは電話でお問合せ下さい。
※銀行振込、または、現金書留でご注文の際には、あらかじめ電話でご確認の上、お申し込み下さい。

■平成2年冬のボーナス一括払いOK!!（12月末）手数料ナシ!!
超低金利クレジットをご利用下さい。

超低金利クレジットO

またまた秋葉原で おなじみの

注目!!
冬のボーナス一括払い
手数料(金利)無料
(平成2年12月末支払いをご利用下さい。)

モデム(AIWA) 50台限定 (送料¥1,000)
PV-A24MNP5 (定価¥54,800)
●MNPクラス5
●2400bps
限定特価¥26,500
(送料・消費税込¥28,325)

8/15〜9/15

●お近くの方は
●本体単品で
●ビジネスソフト

CYBER STICK
●CZ-8NJ2
(定価¥23,800)
超特価!!
¥18,500 (送料・消費税込¥19,570)

X68000シリーズ専用
MIDIインターフェースボード
SX-68M(サコム)
(純生コンパチ)定価¥19,800
特価¥16,480
送料・消費税込み!

X-1ターボZⅢ 特別ご提供品!! 台数限定
●CZ-888C+CZ-860D+M-2HD(10枚)
定価¥269,600▶特価¥164,800
(ボーナス併用も有りますTEL下さい)
・ジョイカード
・ゲーム3種
・パソコンラック(A)3段
プレゼント中
送料消費税込み!!

12回	24回	36回	48回	60回
14,400	7,600	5,300	4,100	3,400

ジョイスティック 送料¥500
●X-1PRO
定価¥9,500▶特価¥7,800
●ASCII STICK
定価¥6,800▶特価¥5,500

P&A超低金利クレジットをご利用ください!!

NEW X68000EXPERTⅡ/Ⅱ-HD & PROⅡ/PROⅡ-HD & SUPER-HD (送料・消費税込)

EXPERTⅡ
セットでお買い上げの方に、
●ディスケット10枚
●ジョイカード2ケ
プレゼント中!!

EXPERTⅡ

セット	定価	特価	12回	24回	36回	48回	60回
(A)セット:CZ-603C+CZ-604D	¥432,800	(価格はお電話下さい。)	?	?	?	?	?
(B)セット:CZ-603C+CZ-605D	¥453,000	(価格はお電話下さい。)	30,200	15,900	11,000	8,500	7,100
(C)セット:CZ-603C+CZ-613D	¥473,000	(価格はお電話下さい。)	?	?	?	?	?
(D)セット:CZ-603C+CU-21HD	¥486,000	(価格はお電話下さい。)	?	?	?	?	?

EXPERTⅡ-HD
セットでお買い上げの方に、
●ディスケット10枚
●ジョイカード2ケ
プレゼント中!!

EXPERTⅡ-HD

セット	定価	特価	12回	24回	36回	48回	60回
(A)セット:CZ-613C+CZ-604D	¥542,800	(価格はお電話下さい。)	?	?	?	?	?
(B)セット:CZ-613C+CZ-605D	¥563,000	(価格はお電話下さい。)	37,700	19,800	13,700	10,600	8,900
(C)セット:CZ-613C+CZ-613D	¥583,000	(価格はお電話下さい。)	?	?	?	?	?
(D)セット:CZ-613C+CU-21HD	¥596,000	(価格はお電話下さい。)	?	?	?	?	?

PROⅡ
セットでお買い上げの方に、
●ディスケット10枚
●ジョイカード2ケ
プレゼント中!!

PROⅡ

セット	定価	特価	12回	24回	36回	48回	60回
(A)セット:CZ-653C+CZ-604D	¥379,800	(価格はお電話下さい。)	?	?	?	?	?
(B)セット:CZ-653C+CZ-605D	¥400,000	(価格はお電話下さい。)	26,800	14,100	9,700	7,600	6,300
(C)セット:CZ-653C+CZ-613D	¥420,000	(価格はお電話下さい。)	?	?	?	?	?
(D)セット:CZ-653C+CU-21HD	¥433,000	(価格はお電話下さい。)	?	?	?	?	?

PROⅡ-HD
セットでお買い上げの方に、
●ディスケット10枚
●ジョイカード2ケ
プレゼント中!!

PROⅡ-HD

セット	定価	特価	12回	24回	36回	48回	60回
(A)セット:CZ-663C+CZ-604D	¥489,800	(価格はお電話下さい。)	?	?	?	?	?
(B)セット:CZ-663C+CZ-605D	¥510,000	(価格はお電話下さい。)	34,100	17,900	12,400	9,600	8,100
(C)セット:CZ-663C+CZ-613D	¥530,000	(価格はお電話下さい。)	?	?	?	?	?
(D)セット:CZ-663C+CU-21HD	¥543,000	(価格はお電話下さい。)	?	?	?	?	?

SUPER-HD
セットでお買い上げの方に、
●ディスケット10枚
●ジョイカード2ケ
プレゼント中!!

SUPER-HD

セット	定価	特価	12回	24回	36回	48回	60回
(A)セット:CZ-623TN+CZ-604D	¥592,800	(価格はお電話下さい。)	?	?	?	?	?
(B)セット:CZ-623TN+CZ-605D	¥613,000	(価格はお電話下さい。)	?	?	?	?	?
(C)セット:CZ-623TN+CZ-613D	¥633,000	(価格はお電話下さい。)	42,700	22,500	15,500	12,100	10,100
(D)セット:CZ-623TN+CU-21HD	¥646,000	(価格はお電話下さい。)	?	?	?	?	?

■P&A恒例サマー大バーゲン祭
開催中!!
◎電話にて、ドンドンお問合せ下さい!!
クレジット表には、出せないほどの価格です。
メーカーさん、ご免なさい。
ユーザーの方には大歓迎されそうです。
今がチャンスです、ハイ。

X68000シリーズ 〜P&Aスペシャルセット=限定誌上販売!!

台数限定 送料、消費税込み
セットでお買上げの方に、
●ディスケット10枚 ●ジョイカード2個 プレゼント中

EXPERT
- ●CZ-602C+CZ-612D……定価¥475,800▶特価¥306,000
- ●CZ-602C+CZ-604D……定価¥450,800▶特価¥300,000
- ●CZ-602C+CZ-605D……定価¥471,000▶特価¥320,000
- ●CZ-602C+CZ-613D……定価¥491,000▶特価¥336,000
- ●CZ-602C+CU-21HD……定価¥504,000▶特価¥338,000

EXPERT-HD
- ●CZ-612C+CZ-612D……定価¥585,800▶特価¥375,000
- ●CZ-612C+CZ-604D……定価¥560,800▶特価¥369,000
- ●CZ-612C+CZ-605D……定価¥581,000▶特価¥386,000
- ●CZ-612C+CZ-613D……定価¥601,000▶特価¥403,000
- ●CZ-612C+CU-21HD……定価¥614,000▶特価¥407,000

PRO-HD
- ●CZ-662C+CZ-612D……定価¥527,800▶特価¥339,000
- ●CZ-662C+CZ-604D……定価¥502,800▶特価¥333,000
- ●CZ-662C+CZ-605D……定価¥523,000▶特価¥352,000
- ●CZ-662C+CZ-613D……定価¥543,000▶特価¥368,000
- ●CZ-662C+CU-21HD……定価¥556,000▶特価¥372,000

●本広告の掲載の商品の価格については、消費税は含まれておりません。 ●営業時間=平日AM10:00〜PM7:00、日祭AM10:00〜PM6:00

ADVANCED 2D GRAPHICS続論

8月号で発表したアンチエリアシング用のグラフィック関数の応用例と新しい関数群。65536色でより滑らかな色調表現と，手描きや従来のグラフィック処理では難しかった自然物の表現に挑戦する。65536色のグラフィックはまだまだ可能性を秘めていることがわかるだろう。

下描きをスキャナで読み込み，点列エディタを使って上からなぞっていく。そうして滑らかな画像データが手軽に作れるようになる。

高品位グラデーションの例。65536色をフルに使えばこの程度の表現力は当然。内部処理は24ビットなので非常に滑らかな変化となっている。

画像取り込みした自然画にランダムフラクタル処理をかけてみたところ。軽くかけると絵画調の表現となる。

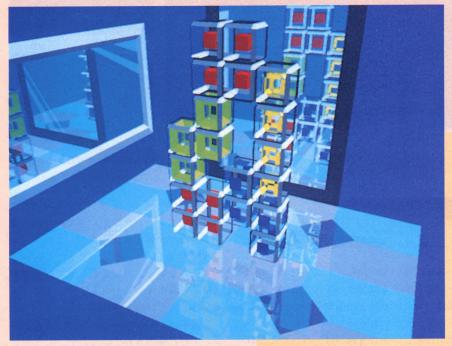
グラデーションで描いた絵をランダムフラクタル処理で加工したもの。雲のような自然な模様ができる。同じ系例のフラクタルでも，フラクタルのかけ方をだんだん強くしていくと，絵もどんどん違う形に変化する。特殊効果としても面白い。

カラーイメージスキャナ(WD-05HS)での取り込み例。なお，アダプタは一部手直しが必要。

DōGA・CGアニメーション講座

今月は皆さんからのご要望にお応えして画像を大映しで載せてみました。これで細かいところの処理や，全体的なバランスなどがよくわかると思います。
　DōGAの連載も1年を過ぎ，皆さんも基礎的なことは把握できたと思いますので，今月はちょっと高度な技術を紹介してみました。

画像2　遠くもはっきり，くっきり，わあキ・レ・イ

画像3　空気遠近法なら遠くはかすんでリアリティが出る

画像1
床への映り込みの例。位置関係をよく見てください

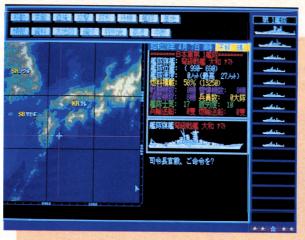
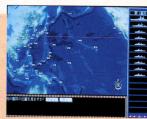
モデラー高津のアップデータ
いや，なんとも，夏ですねぇ

画像4
ゴジラの親子はなかよしこよし，てなカンジかな（ちがうって）

モデラー高津のLOGIN　うにゃらうにゃらと動く。長く見ているとだんだん気分が……

SOFTWARE INFORMATION

SOFTWARE INFORMATION

今月は暑さのせいかゲームのほうもちょっとひと休みってカンジかな。とりあえず，今回は先月号で間に合わなかったゲームを中心に紹介していきます。てなことで，今月もヨロシク！

提督の決断
光栄の新作は太平洋戦争が題材のシミュレーション。マルチシナリオ方式を取り入れている。歴史の勉強ができちゃいそう。

話題のソフトウェア

あ～，寒い寒い。なんでこんなにクーラーがききすぎるほどきいてんだろ，この建物は。おかげで外との温度差がやたら激しくってカゼひいちゃったじゃないの，もう。確かに新しいからきれいはきれいなんだけどね，このビル。

さて，今月はスペースが少ないぞー！　全部紹介しきれるかなあ。それじゃ，スペースも少ないことだし，ぐだぐだいってないで，さっさと進めていきましょうか。夏休みも残り少ないし……ね（だって宿題全部終わった人って少ないんじゃない？　そろそろ焦ろうね）。

トップバッターはお馴染み光栄のシミュレーションゲーム**提督の決断**。これは太平洋戦争を題材にしたもので，プレイヤーは日本軍か連合国軍のどちらかを選択し，太平洋の利権を巡ってドンパチするわけです。当然シミュレーションゲームだから頭を使う。どんな作戦を立てるかが，このゲームの攻略のポイント。友達と２人でも遊べるのもうれしいですね。これはすでに発売中。来月号で詳しくやるつもりです。さて，キミならどんな決断を下す？

２番手はT&Eの**ルーンワース**。このゲームもすでに発売されているので，もうプレイした人もいるはず。プレイヤーの行動によってエンディングが変わるマルチエンディング方式を採用したRPG。これからじわじわと人気を集めてきそう。これも来月号で詳しくやります。楽しみにしてて。

それではザインソフトの**バルーサの復讐**。このゲームももうすぐ発売される予定。画面を見るとザインらしさがよく伝わってきますね。このゲームはトリトーン・ファイナルの続編です。

さあさあ，以前からみんなの興味をひいていたシステムサコムの**闇の血族**，これも

泰若自然，ポピュラス！

順位	タイトル	(前回順位)
1	ポピュラス	1
2	ダンジョンマスター	4↑
3	グラナダ	2↓
4	天下統一	5↑
5	クォース	―初
6	ギャラガ'88	―初
7	スーパーハングオン	6↓
8	ソーサリアン	10↑
9	トンネルズ＆トロールズ	―初
10	ワンダラーズ・フロム・イース	3↓

相変わらずポピュラスがトップです。２位ダンジョンマスターの票の2.83倍というからお話にならない。来月からプロミストランドの票も入ってきたらどうなってしまうのやら。

しかし，ポピュラスを僅差で追っているソフトが１本だけあります。それは“SX-WINDOW”だっ！　「安い」「ヘタなゲームより楽しめる」「遅い（？）」などなどの声が聞かれます。ほかにゲーム以外ではＣコンパイラ，Z'sSTAFF PRO-68Kの票も多いですね。

５，６位はそろってゲームセンターからの移植が初登場。クォースはゲームセンターでハマっていた人のハガキが多いのが特徴で，対戦が楽しみという声も。対してギャラガ'88はグラフィックがいい，手軽に楽しめるということですから，ゴテゴテせずにシューティングの王道をゆく姿勢が，好印象を与えたということでしょうか。

トンネルズ＆トロールズは，豊富なテキストと数多くのイベントが飽きさせないという声で９位にランクイン。しかし上は強豪ぞろいでツライかな。

ところで，ソーサリアンがランクアップしてるけど，このハガキの大半に「ランクから落ちそうだから」て書いてあるぞ。うーむ。おそるべきパワー。今後はイースが落ちそうだから，ファンのみんな頑張ってハガキを送ってくれたまえい。（浦）

ルーンワース
マルチエンディング方式採用のRPG。もちろんそれ以外にバッドエンディングもあるけどね。口の悪い勇者っていうのも珍しい。思わず笑っちゃいます。操作性はなかなかGOOD。とっつきやすいゲームといえるでしょう。

バルーサの復讐

闇の血族

PINBALL・PINBALL

雀豪2

とうとう“1”が発売されました。超能力を持つ少女魅由が，殺人事件を解決していくアドベンチャー。取り込み画像を駆使したオープニングやビジュアルシーンは圧巻ものです。エンディングが終わると，9月に発売される予定の完結編の予告まで入っていたりして，なんとなくトクした気分になれますヨ。

新規参入会社日本ソフテックからは**PINBALL・PINBALL**が発売される予定。タイトルを見てのとおりピンボールゲームです。サンプル版を見る限りではなかなかよくできているようす。期待しましょう。

で，ダンジョンマスターがいまだ好評を得ているビクター音楽産業からは，**雀豪2**が発売される予定。推論型人工知能を搭載したことで麻雀ファンの間で人気だったが，今回のバージョンアップでますますその傾向に拍車がかかりそう。なんたって今回は「強知能版」。一層本格的な麻雀が楽しめるとあっては，麻雀ファンでなくても興味津々てところでしょう。8月中に発売の予定です。

先月号でも紹介した**ワールドコート**。いよいよSPSから発売されました。このゲーム，最初はなかなかタイミングを合わせるのが難しいんだけど，慣れてくるとエースが決まったりして壮快そのもの。サンプリングでスコアを喋ってくれるのもなんとなくうれしいし。このゲームは要2メガです。

で，新規参入のポニーテールソフトでは，パズルゲーム**ユニオン**が快調に開発されているもよう。画面上にある同じ牌と牌を合体させることによってクリアできるといったパズルゲーム。いつも同じ牌を見るんじゃ飽きちゃうから，牌の種類はフルーツ，動物，英字，昆虫などといろいろ用意されています。すべてマウスで操作できるのもX68000用らしくていいですね。

さてと，発売はまだ先だけれど着々と進行している2つのゲームの画面を紹介します。まずはシステムサコムの**ジェミニウイング**。そろそろ遊べるものができてきました。上がりが楽しみですね。そしてM.N.M. Softwareの**ペルセウスの冒険**。これは横スクロールタイプのRPG。もう少ししたら詳しくお伝えできそうです。

そのほかウルフ・チームではX68000専用仕様のアクションゲーム**アクシス**を開発中だし，ヘルツでは，アクションゲーム**ダイナマイトデューク**を移植開発中だし，**遊撃王II**はもうじき発売されそうだし。

なにはともあれ活気のあるゲーム界。来月はいったいどんなゲームが出てくるやら。それじゃ，また来月をお楽しみに。

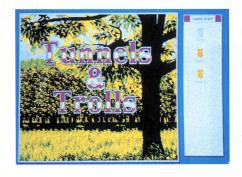
ワールドコート

ユニオン

ジェミニウイング

ペルセウスの冒険

THE SOFTOUCH

●TUNNELS & TROLLS

テーブルトークの興奮をX1でも!

Mizuno Kazuo
水野 一雄

かの有名なテーブルトークRPG「TUNNELS&TROLLS」がパソコンゲーム化され,X1およびX68000版でも発売されました。特にX1ユーザーの方々には興味津々かもしれないけど,お気に召しますかな……?

X68000用 5″2HD版3枚組 9,800円(税別)
X1/turbo用 5″2D版12枚組 9,800円(税別)
スタークラフト ☎03(988)2988

ついに,「TUNNELS&TROLLS」(以下T&T)が出ました。テーブルトークでプレイしていた私はすっごく嬉しい。X68000はディスク3枚組ですが,X1ではなんと12枚組です(超大作ともいえますが,いやな予感がするでしょ)。それでは,さっそくいってみましょう。

テーブルトーク「T&T」

皆さん,テーブルトークはご存じでしょうか。簡単にいうと,数人で対話をしながら進めていくRPGです。映画「E.T.」の最初で子供たちがピザを注文しながら遊んでいたのがそうです(あれはD&D,ちなみに,あの「ウィザードリィ」はD&Dのダンジョンでの冒険を簡略化しゲームにしたもの。知ってた?)。

さて「T&T」ですが,これは本格的でありながら,システムやルールはすごく簡単(柔軟性がある)なんです。ほかのRPGはちゃんとしたパッケージにルールブックなどが入っているものが多いのに対して,これは必要最低限のルールが書かれた文庫本1冊(消費税実施前に680円だった)で出ています。そして「T&T」には,ソロアドベンチャーなるものがあります。これはいわゆるゲームブックで,「T&T」の世界(もちろん,ドラゴン大陸が舞台)をひとりでも遊ぶことができます。これも,簡単なシステムの恩恵といえるかもしれませんね。興味のある方はやってみるとよいでしょう。

パソコンゲーム「T&T」

このゲームがどんなものか,システム,キャラクタ,アイテムに分けて紹介していきましょう。

まずはゲームのシステムについてですが,ウルティマ型の本格的ロールプレイングゲームです(これは見ればわかるか)。あっ,ウルティマ型って知っていますか。これはゲームの世界(マップ)を上から見下ろすタイプのRPGの総称です(対するものにウィザードリィ型があり,これは3Dダンジョンタイプのものですが,両方ともすでに死語?)。単に上から見下ろしていても芸がないので,このゲームではアドベンチャーフィールド(マップ)を斜め上方から見下ろすという方法をとっています。初めのうちは違和感がありますが,慣れると結構見やすいと思います。

パーティの移動方法もテーブルトークらしくなっていて,「普通に歩く」「探索しながら歩く」「走る」「登る」など,いろいろとあります。探索していれば罠に掛かりにくく,些細なことにも気づく確率が増しますが,やっぱりそれなりに時間を消費します。走っていれば速く移動できますが,危険にあいやすくステータスも回復しないということになるので,TPOに応じて使い分けが必要です。

戦闘はタクティカルコンバット方式です。戦闘時のメニューには,「Auto Fight」「Manual Fight」「Run」「Control」があり,「Auto」は個々のキャラクタが手にしている武器で自動戦闘します。絶対勝てる相手のときには便利でしょう。「Manual」は個々のキャラクタの動作を決定できます。戦闘は敵,味方の区別なく敏捷性(SPD値)の高いキャラクタから順に行動します。

動作中のメニューには「攻撃する」「魔法を使う」「アイテムを使う」「飛び道具を使う」「体当たりする」「防御する」があります。ここで面白いのは「体当たりする」でしょう。これは文字どおり相手に体当たりをして,うまくいけば相手をふっ飛ばしたり気絶させたりできます。また注意する点として,「防御する」以外は味方の位置も考えないと,味方にも影響を及ぼしてしまうということ(仲間を攻撃したり,魔法で眠らせたりしてしまう)があります。「Run」は皆さんお馴染みの“逃げ”です。そして,「Control」はメッセージの表示速度の変更など。ここはX1とX68000で少し異なって「アニメーションのオン,オフ」なんてX68000らしいものもあります。

装備を身につけると画面上の姿も変わる

ところで，このゲームでは時間の概念があり，ゲーム進行と密接に関係してきます。この世界では１年は12カ月で四季があり，１週は７日と我々の世界とあまり変わりありません。日付，曜日によって生活が変化するので，プレイヤーの対応次第では冒険に有利になったり不利になったりします（具体的には食料の値段が月により変化するなど)。時間は１日24時間です。時間は商店の営業時間，ステータスの回復などに関係してきます。

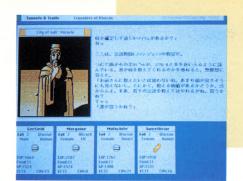
キャラクタについて

やはりRPGなので，定番のキャラクタメイキングがあります。しかし，「T＆T」ではNPC（ノン・プレイヤー・キャラクタ）というものがあり，すでにシナリオ中にキャラクタが用意されているので，それを使ってプレイすることもできます。でも，自分の分身（PC：プレイヤー・キャラクタ）だし，ゲーム中で役に立つときもあるかもしれないので，作成しておいて損はないでしょう。

キャラクタは４人まで作成できます。作成できる種族はHuman, Elf, Dwarf, Hobbitの男女の８種類，職業はWarrior，Rogue，Wizardの３種類です。また，キャラクタの容姿（グラフィックで表示される。Elfの女性はいいが，Dwarfの女性は……）も選択できます。X68000ではきれいですが，X1ではこんなものかなという程度です（まあ，そんなに重要じゃないけどね)。

ところで，テーブルトーク「T＆T」では種族にフェアリーやワーウルフなどのモンスターを選択することもできるし，職業にも魔法戦士がありました。これらはゲームバランスを崩しかねないので実現されなかったのでしょう。僧侶は「T＆T」では扱いません。治療の呪文も魔術師が唱えますから。ちなみに盗賊もある程度までの魔法を唱えられます。

キャラクタのステータスにはST（体力度)，IQ(知性度)，LK(幸運度)，CON(耐久度)，DEX（器用度)，CHR（魅力度)，SPD（速度)，ADD（戦闘修正）があります。ADD以外はお馴染みでしょう。ADDはほかのステータスより導き出される戦闘時のボーナス点です。

キャラクタ作成時の解説はこの辺にしておいてゲーム中のことに話を移します。ゲーム中にキャラクタを呼び出すと，ステータスのほかにAGE（年齢)，Level（レベル)，Hits Taken（防御点)，FOOD（食料)，GP（金貨)，SP（銀貨)，CP（銅

言語を教えてくれる人がいた，誰が習おうか

これが戦闘画面だ！　バシバシ

貨)，健康状態が表示されます。「T＆T」では貨幣が３種類あり，1GP＝10SP＝100CPとなっています。健康状態は８段階あり，X1では文字，X68000ではちょっとしたグラフィックで表示されます。また，作成時に選択した容姿も表示されます。X68000では装備を変えるとこれが変化します（X1の人，残念でした)。「T＆T」には豊富な言語があります。誰もが話せる共通語のほか，エルフ語，ドワーフ語，ホビット語，古代語(モンスターの話す言語)，獣語（動物の話す言語）など十数語あり，いろいろな言語を習得することにより話せる相手が増え，冒険に幅を持たせてくれます。

冒険するにはパーティを組まなければなりません。パーティは４人編成でPC，NPCの混在が可能なので，必要に応じて入れ換えることができます（入れ換えたキャラクタはその場でずっと待機している)。

最後にアイテムについてですが，「T＆T」の特徴は武器の豊富さにもあります。短剣，直刀，曲刀，こん棒，斧，弓，槍などの武器がすべて数種類ずつ揃っています。これだけあると攻撃力の同じものがあるので，「この武器は嫌いだが攻撃力があるからしょうがない」などということがなく，自分の好みで選ぶことができます。

実際にやってみて

テーブルトーク「T＆T」のプレイヤーから見てもかなりいい出来ではないかと思います。テーブルトークとは異なっているところがかなりありますが，それはしかたないといって通り過ぎることができる程度のものだと思います。

X68000版では(編集室で８時間程度遊んでみただけですが)，これといって不満はありませんでした。ディスク交換も起動時に１回のみだし，ハードディスクにインストールして遊ぶこともできます。

X1版は（たぶん）初代でも走るだろうからそれだけでもすごい。でも，不満もあります。ディスク12枚という大作であるから，（怒濤の）ディスク交換は避けられないとしても，１戦闘につき４回（のべ６枚）は歓迎できません。プレイ時間よりもディスク交換，アクセスにかかる時間のほうがはるかに長い気がします。これでは感情移入もへったくれもあったもんではありません。もう少しどうにかならなかったのでしょうか。あと，X1turboなのにグラフィックに漢字を書いているではありませんか。パッケージにある「400ラインモニタ，漢字ROMが必要」とは，どこで活かされているのでしょうか。また，両機種ともマウスを使用できますが，アイコン間が離れていることもあり，移動量が大きくて使いづらいところがあります（マウスを使い慣れていないせいかもしれないが)。この辺も考慮に入れてもらって，次回「クォータースタッフ」に期待したいと思います（移植されるかなぁ……X1に)。

感想

かなり忠実にゲーム化されていますので，テーブルトークの「Ｔ＆Ｔ」を知らない人でも，雰囲気を味わえると思います。だから，細かいことをいってもしかたないと思いますのでいわないことにします。

また，X1でも走ることは大いに称賛すべきことだと思います。X1ユーザーの皆さん，怒濤のディスク交換に負けず頑張ってください。

X68000版	(最高＊10個)
シナリオ	＊＊＊＊＊＊＊
グラフィック	＊＊＊＊＊＊＊
BGM	＊＊＊＊＊
ゲームバランス	＊＊＊＊＊＊
「T&T」らしさ	＊＊＊＊＊＊＊＊＊＊
総合	＊＊＊＊＊＊＊＊＊

X1/turbo版	
シナリオ	X68000と同じ
BGM	同左
ゲームバランス	○
「T&T」らしさ	かなり良い
ディスク関係	涙ちょちょ切れ
総合	ディスクさえ我慢すれば◎

●D-Again

キャラクタは世界をつくる

Komura Satoshi
古村 聡

一部に熱狂的ファンを持っているブロンウィン。そのブロンウィンが活躍する「第4のユニット」シリーズも，はや第5弾。これで第1部「WWWF」編が完結らしいけど，どこまで続くんだろう……。

X68000用　5″2HD版4枚組　8,800円(税別)
データウエスト　☎06(968)1236

やっぱりキャラクタなのだ

うーん，うーん，感動したぞぉ。やれば感動するぞぉと思いながら，やり終わってやっぱり感動するっていうのは作れそうでなかなか作れるもんじゃありません。というわけで，第4のユニットシリーズ第5弾“D-Again”の登場なのです。いや，このシリーズはシナリオと演出が本当にいい！　と，いつもながら感心してしまいます。前作の第4のユニット4“Zerø”ではほとんど詩のような囲み（本人は詩なんか書いた覚えはないんだが……でもあれって本当は2ページ見開きができるくらい分量があったんだよ）を書いてしまい，いまではOh!Xの吟遊詩人とまで呼ばれるようになってしまった私なのでありますが，そうやって道を踏み外す奴が出てくるくらいシナリオ&演出がとてもよいのです。

なんでこんなにシナリオも演出もよく出来ているのか？　それはこのゲームがキャラクタを大事にしているからなんですよね。登場してくるキャラクタをあたかもそこに本人がいるように，細かく細かく心理描写をしていく，そしてキャラクタがひとりでに話を作っていってしまうのではないかというくらいキャラクタが感情豊かになっていく……ということです。

前作“Zerø”ではブロンウィンのキャラクタがかなり形となって出来てきたんですよね。そして，今回はもうひとりの少女の物語なのです。

あらすじなど……

話は前作“Zerø”のエンディングから。ブロンウィンは統合軍の特務査察官に任官されたのです。そして2週間の研修の後，彼女は局長から特務機構長官護衛の指令を受けます。命令を遂行すべくパリへと向かうブロンウィン。

しかし，パリに着いたら着いたで，もうパニック，結構前途多難なブロンウィンだったりするのです。まー，いっしょに仕事をするSクラスエージェントのオージスっていう男がとんでもない奴。3度の飯より飯が好き，酒は昼間からかっくらう，ブロンウィンを抱きかかえて地下鉄に乗り込むわ，挙げ句の果てに自分のことを愛情を込めて“オージー”って呼んでくれだって（……わし，こいつ嫌い！）。本当に任務が遂行できるのかなあって感じなのです。

ジュネーブまでの特務長官の護衛の途中，列車の中でブロンウィンとともに護衛の任についたオージスが話しかけてきます。

「さて，問題です」

「なによ」

「次のヒントに当てはまる人は誰でしょう？

1．目は猫目で赤い瞳をしています。

2．髪の毛の色は青です。

3．その髪型はポニーテールです」

「ダルジィ!?」

「さっきホームにいたぜ」

「！」

ダルジィがこの列車に乗っているのか？　WWWFは崩壊し，もはや機能していないはずではなかったのか。ダルジィは何の目的でこの列車に乗っているのか？　はたしてダルジィの逆襲はあるのか？　WWWFは本当に崩壊したのか？　そしてブロンウィンの前に立ちふさがる新たな敵とは……？　ブロンウィンの運命は……！

そう，やっぱり今回のストーリーは第2話での登場以来ずっとブロンウィンと戦い続けてきた，永遠のライバル，ダルジィのお話なのです。

第4のユニット4“Zerø”ではブロンウィンに人格が出来てきてひとりで歩き始めた，ということを見せた話だったのですが，考えてみるとやっぱり，ブロンウィンというキャラクタが出来上がっていくうえでいちばん影響を受けた（お互いに）のがこのダルジィなんですよね。永遠のライバル，ダルジィ。ブロンウィンを光とすれば彼女は影といえるかもしれません。しかし，光と影は表裏一体。ブロンウィンというキャラクタと同じように，ダルジィもかなりキャラクタが出来上がってきたんです。その証拠に今回じつはダルジィはこんな台詞を言うんです。

「後悔などあるものか。後悔など……」

これがクライマックスの感動へとつながっていくんです。前作のエンディングでブロンウィンが見せたあの笑顔。そして，今

この人に仕事を頼まれちゃうのだ

回見られるダルジィのさまざまな表情。ブロンウィンが「優しさ」という言葉で表せられるとすれば，ダルジィのそれは「強さ」なのですが，あの台詞が彼女のガラスのような硬く，しかしもろい強さなのです。

ま，とにかくストーリーは皆さん自身でじっくりとこのゲームをやってみるのがいちばんでしょうね。ぜひ，この感動のストーリーを味わってください。

なんじゃ，こいつは……ようし！

というふうに，こいつだって戦える。だけど……

システムについて

あんまりストーリーの話ばっかりしてもしかたがないので（下手すりゃネタばらしになってしまいかねないし，それだけは絶対したくない……）ゲームのシステムの話をしましょう。とはいっても，善ちゃんと，

「あ，PCMでサンプリングやってるね」

「そーなのー」

「だって，これハンドクラップじゃん」

「そーなんだー 」（「あーくしゅ」のじぇだか，おまえはっ！ あれも面白かったけどね）

という間抜けな会話が成立してしまうくらい，音楽にもグラフィックにもセンスのない私だからしてたいしたことはいえないんだけどね。

画面写真を見るとわかるでしょうけど，この“D-Again”は画面レイアウト，プルダウンメニューなどゲームのシステム自体は“Zerø”そっくりなのです。

別にそれ自体は間違いではないと思うんですよね。だって，このシステム自体とてもよく出来てるし，このゲームを買う人の中のかなりの人が“Zerø”を買ってるでしょうからみんなが違和感なくゲームを進めるにはもっともいい方式だと思うんですよね。それに演出的にも4でブロンウィンの心理描写，5でダルジィの心理描写……と同じようなことをやりたいのであればとてもいい方法だと思うんですよね。私はこれでよいと思います。

ただ，唯一残念だったのが，あの第1話（第4のユニット）から採用され続けてきた

飛行機の中でたたずむブロンウィン

“デフォルメキャラ戦闘モード”（勝手に名付けてしまったが，2頭身のキャラクタがちょこまか動いて戦う戦闘モードのこと）がなくなってしまったのですよね。戦闘モード自体がなくなってしまったわけではなくて，2頭身キャラがちょこまか動くのがなくなって，代わりにメッセージにダメージの数値が表示されるようになったんですけどね。

うーむ，確かにあの戦闘モードは作るのが面倒だということはとてもよくわかるんだけど，でもあれってとってもかわいくて楽しかったのになー（ちなみに私は第1話の戦闘場面を見てプロレスゲームかと思ったことがある。なつかしい思い出だ……）。それに相手のダメージが数字で出ると敵の弱点がすぐわかっちゃって面白くないんですよねー。次回作ではどうにかしてくださいね。データウエスト様。

跳べ！ アニメーション効果

さてさて，今回のシステムの最大の特徴。今回のシステムと前回のとで決定的に違うのはゲーム中にアニメーションがある，ということなのです。

“D-Again”はFM TOWNSではX68000より2カ月ほど先に出ているんですが，「第4のユニットは今度からアニメーションがある。あのブロンウィンが画面の中で動き回る」という話を聞いて，いてもたってもいられなくなり本当にブロンウィンを心から愛しているのならTOWNSを買うしかない！ でも，そんな金はないし……。んが，だがしかし，だがしかし。はっきりいってそれはいらん心配でした。

「よ，よくも！ ぶち殺してくれるっ！」

化けもんがガバっと迫ってくる！

ぱっと跳ねるブロンウィンたち！

そしてスパッと切れる服！

ちゃーんと，飛びかかる，跳ねる，切れるといったアクション表現がバリバリにアニメーションしてるんです。いやー，ちゃんと動くじゃないですか，はっはっはっ。すごいですよぉ。躍動感いっぱい。それもクライマックスでのぐいぐいと引き込むようなストーリーの流れとあいまって完璧といえる演出。言葉でなんかとても表現できません。

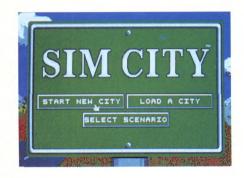
これだけの技術をぜんぜんウリにするでもなく（TOWNS版の広告にはありましたが，X68000版ではパッケージにすら書いてないんですよ，ちょっと心配しちゃった）惜しげもなく，しかも効果的に使ってしまう。展開もスピーディに感じさせてくる。完璧，これ以上何を求めようかという演出です。いやはや感動のアニメーションなのです。はっはっはっ。

今回の“D-Again”で5作目になる第4のユニットなのですが，コレークターズアイテムといえるアドベンチャーゲームの代表作といっても，もう誰も異論を唱える人はいないでしょう（いても聞かないふりをするもん）。ついにデータウエストさんも「このデータウエストある限りこのシリーズはずっと作り続ける！」と宣言してしまったくらいのすごい力の入れようなのです（うむうむ，よいことじゃ）。ぜひともデータウエストさんにはがんばってこれからも第4のユニットシリーズを作っていってほしいですね。

思い入れ

パッケージではブロンウィンは髪を結わえている（注：日本髪のことではない。束ねてるっていうのかな？）のだけど，なぜかゲーム中には出てこない。うーん，かあいいなー。かあいいなー。かあいいなー。ぜひゲームの本編の場面でもあの絵ほしかったなー。あと，最近遊びの部分がちょっと少なくなっちゃった気がするんですけど……あってもいいんじゃないですか，もう少し。ま，でもいいゲームです。どんどんこの調子でいってほしいな（しかし，WWWFの最後が，ショッカーのあとにはゲルショッカーってパターンとはねぇ……）。

評価

大変よくできました

●シムシティー

渋滞は都市の常，メルトダウンは神の業

Ogikubo Kei
荻窪 圭

待ちに待ってたシムシティーがいよいよX68000に登場。ポピュラスでは神サマだったボクたちは，今度はもうちょっと身近な市長さんに早変わり。さてさて，それではメガロポリスでも目指すとするか。

東京には首都高という一般道から隔離された，くねくね曲がるわ，重なるわ，間違えるとどこへ連れて行かれるかわからないわの芸術的な道路がある。ビルの谷間やら川の上やら堀の中やら，トッピーなところを走ったりして面白いことは面白いのだが，実によく渋滞する。車の流れより芸術性を重視した設計をしたせいなのだが，だからといって一般道を走ると，これまた信号がいっぱいあって，歩行者がたくさんいて，いろんなところで工事をやっていて，不法駐車がたくさんあって，警察は不法駐車をたくさん取り締まって，車の数を減らしたいのだがあんまりやりすぎると「経済大国日本」を支えている自動車産業が怒ってしまうので実行できないでいて，首都高以上に時間がかかったりする。

しかしである。この前，私は「そーだよな，どこがどう渋滞するかなんて，どーろ作るときにはわかんないもんな」なんて，道路つくってる役人に同情してしまったのだ，不覚にも。それもこれも，このシムシティーがいけないのである。私はシムシティーやってるときは市長でも，普段はなけなしの収入からひいひいいいながら，所得税や住民税や国民健康保険料やら国民年金を支払っているビンボーで善良な市民なのだ。官僚や政治家の無策を許してはいけないのだ。それに，シムシティーの市長はツインタワーの庁舎の高いところに広いスペースなんて持っていないのである。せいぜい，猫の額の6畳間だ。

究極の箱庭ゲームであること

シムシティーは都市シミュレーションゲームである。ポピュラスやA列車で行こうと同じくリアルタイム・シミュレーションゲームである。私は吉田幸一氏の呼称が気に入っているので「箱庭型ゲーム（Oh!X5月号P40参照）」と呼んでしまおう。「箱庭療法」を英語でサンドプレイ（SAND PLAY）というので，サンドプレイゲーム，ってなわけだ。ロールプレイングゲームだって心理療法の一種である役割演技（ロール・プレイング）からきているのだから，サンドプレイゲームがあったって，いーじゃないか。

さて，リアルタイムでポピュラスやA列車を思い起こすようなゲームであるからして，箱庭の中でポコポコと人が増えたり生活したりするのを見て楽しむのである。ポコポコと人を増やすには，住みよい都市を作らねばならない。そのためにはマウスを使って何もない土地を区画整理する。

各区画は原則として3×3のマスが1単位であり，居住地区，商業地区，工業地区の3つが基本である。さらに警察と消防署も都市には必要だ。

都市が発展するには遠距離の交通，交易のための拠点も必要で，港と飛行場を置くことができる。飛行場は大きいので5×5マスである。ついでに娯楽も必要だということで，古代ローマのようにスタヂアムも建設する（造らないと，住民が要求する）。

それでもって，交通網の整備も忘れてはならない。道路と鉄道の敷設が可能だ。駅を作る必要はない。A列車のように列車事故が起きることもない。

居住地区，商業地区，工業地区の3つについては，プレイヤーは区画整理するだけで，建物を建てるのは市民の仕事である。区画を作れば人が集まる，なんてことはなくて，治安の悪いところに人は住まないし，職場のないところに人は住まないのだ。

操作はすべてマウス。ウィンドウばしばし。MusicstudioやMu-1のように1024×512の画面を使っていて，画面からはみ出る分についてはすいすいスクロールする。ダンジョンマスターやポピュラスのように，

実にアメリカンなのがポイントだ

シムシティーには農地がない。なぜなら，西欧の「都市と農村を区別した都市論」の上に成り立っているからだ。日本では昔から都市と農村なんていう区別はあまりなくて，今でも都市の中にポツンと小さな農地があったりするが，アメリカでの農業はでっかい田園地区にどばっと土地を持ってやるのが基本だからだ。で，シムシティーには農地はないのである。食い物はどーしてるんだろう，と思うのだが，きっと，工場でつくっているのだろう。

スタヂアムではアメリカンフットボールをやっている。これもアメリカンだ。サッカーならばヨーロピアン，相撲ならばジャパニーズ，ローラースケートをやっていればジャニーズだ！ ちなみに，スタヂアムでは49ers（フォーティーナイナーズ，などといううっとうしい読み方をする）がBearsに7－10で負けている。

居住区は突如として教会に化けることがある。これもアメリカンだ。なぜなら，日本だったら，「人々が自分で寺を作るより，政府が勝手に作るのが近代の基本」だからである。異論はあるだろうけど，いいや。

さらに，警察からほんのちょっと離れると「治安が悪くなる」のもアメリカンである。さらに，治安が最悪のマンションでも人が住んでいたりするところもアメリカンである。

それに，電気がエネルギーの基本，ってのもアメリカンかもしれない。まあ，なんにもない土地にボコンと都市を作ってしまうところも，入植の歴史を持つアメリカンなところだな。

X68000用　5"2HD版　9,800円（税別）
イマジニア　☎03(343)8911

「なぜか狭い画面」を味わうことはないわけである。以上，ゲームの説明終わり。

なお，コンピュータがジェネレートした土地に都市を作って楽しむモードと，状況設定されたシナリオの上で都市復興や再建を目指すシナリオモードがあるんだけれど，私の趣味の問題でシナリオモードはやんなかったので了承してね。

都市開発の苦労と喜び

1900年。最初は何もない。空き地と森林と川（あるいは湖）があるだけである。プレイヤーはそのマップが気に入ったら，おもむろにポン，と発電所を建てる。発電所には火力と原子力があって，火力発電所は港の近くにってなことはない（そんなのは燃料を外国に頼っている日本だけか）。それから，原子力発電所だとメルトダウンすることがあるらしい。が，私はとりあえず「災害なし」モードで遊んでいるので関係ない。1900年に原発？　なんて野暮なことはいわないように。

で，居住区をペロっと決める。食う寝るところに住むところがないと話にならない。でもって，発電所から送電線を引っ張る。続いて，職住接近でも離れててもいいが工業地区やら商業地区やらを指定する。うまくいくとバラックが高層マンションになったり，西友が有楽町マリオンになったりする。恍惚の瞬間だ。そして，道路やら線路やらで結んでいけば街ができていくわけ。

この時点で重要なのが，どんな都市を作りたいのかというコンセプト。歴史が語るとおり水を中心に考えるシティ。ヨーロッパ風に放射状の街（もっとも，道路は縦か横なので，きれいな“すべての道はローマに通じる”は作れない）。平安京や長安のような碁盤目の都。

ひとつ，歴史が与える教訓を記しておこう。それは，ブラジルの首都“ブラジリア”とオーストラリアの首都“キャンベラ”だ。この２つの，山か森か知らないけれど何もないところを切り開いて作った人工的な都。結局，人々は整然とした近代都市より，その郊外に建てたバラックに好んで住みついたという。日本でも幕張あたりがそういったハイテクタウンのグロテスクさをかもし出しているようだ。親しみを感じない都市って，いらないよね。

とりあえず考えるのは，工業地区は治安が悪くなりやすく，空気も悪くなりやすいので，居住区との距離を考えることとか，商業地区は人が集まるところに作るとか，水辺に近くて環境の良いところに人は住みたがるということくらいだ。

怪獣が通ったあとは火事になる

評価表は市民の不満のバロメータ。おーこわ

公園も作れるが，まとまった広さの公園をでんと（セントラルパークみたいに）作るか，居住区の中に小さなのをちょこちょこ作るか（高輪３丁目児童公園みたいに）は好きずきだ。

なんて，書くと，楽みたいだけど，実は“災害なし”モードで成長を「うふうふ」としているからであって，ノーマルモードにするとたちまち「えらいこっちゃ」なのである。シムシティーを走らせたまま，「飯でも食ってくるか」しようものなら，火事で都市1/3くらいが丸焼け，ケチって消防署を作らなかったばかりに，燃え広がる一方，仕方ないから火のまわりを破壊して延焼を食い止める，なんて羽目に陥るのだ。

ひと通り，でかい都市を楽しんだら，災害ONでやってみようと思っているので，「軟弱ものめ！」と責めないでおくれ。

ちなみに，任意の災害を起こすこともできる。飽きたら，火災や洪水，飛行機事故から竜巻，怪獣まで好きな災害を起こそう（これはポピュラスか？）。

市民の気持ちの汲み取り方

当たり前だが，都市を大きくするにはコツがある。たとえば，治安とか，大気汚染とか，道路とかだ。道路が多いと車が増え，酸性雨が降ったりして大気は汚染される。

治安をよくするには警察を作り建物は密集させないように。大気汚染には公園や鉄道（鉄道は大気を汚染しない）を増やす。

収入は税金なので，人が減ると困るし，無駄使いすると都市の維持に困る。

“評価”コマンドを見てみよう。住民が何に不満を持っているかや，今の人口がわかる。

“グラフ”を見てみよう。各データの年間推移がわかる。

“マップ”を見てみよう。犯罪発生率や警察管轄区域，大気汚染などの分布図が見られる。

そして，　にたにた笑いながら,都市は未来を目指すのだ。未来都市には２種類ある。「ブレードランナー」な都市と「宇宙戦艦ヤマト」な都市だ。いま私がそう決めた。どっちを目指すも君の自由。そーうまくはいかないけれど。

シムシティーもはまると病気になるので，それだけは注意しよう。街を歩きながら，「ここは商業地区」だな，とか「お，渋滞はなんとかしなきゃ」などと気分は市長になってしまうのだ。

ダンジョンマスター，ポピュラスに続く“危険なソフト”であることは保証する。

参考文献

1)トポスの知　河合隼雄・中村雄二郎・他
2)図説　都市の世界史　レオナルド・ベネヴォーロ著　佐野敬彦・林寛治訳　相模書房
3)ロココ町　島田雅彦著（今月の「新刊書案内」参照のこと）

総評

何もいうことはあるまい｡「われわれは箱庭療法的な表現を，実は日常生活の中でも欲していたんじゃなかろうか（中村雄二郎)」ということかもしれないし，すぐに都市計画とか都市論をぶち上げたがる計画好きのせいかもしれない。アートディンクにもチャンスはあったのに，A列車のあと，A列車IIなんて作ってしまった。シムシティーの方向へ進めばよかったのに，と思う。そうすればジャパニーズな箱庭都市ゲームで遊べたのだ。ポピュラスといいシムシティーといい，さすが西欧人は「モデル化」がうまい!!!というところかもしれない。

都市型ゲームはまだまだいける。都市をモデルにシューティングしたのがグラナダであり，そのままシミュレーションにしたのがシムシティーだ。「都市と箱庭」。これがポイントだよ，各ソフトハウスさん。

５段階評価

箱庭度：★★★★★
都市計画度：★★★★★
1/8計画度：★★★★
アバウトな操作でOK度：★★★★
ウィンドウバシバシ：★★★★
民のカマ度：★★★★★
PCM効果音：★★★★　　（ただし，評価版）

シンプルだからこそ飽きないゲーム

Nishikawa Zenji
西川 善司

ゲームの少なかった10年前に登場したギャラクシアン。そのギャラクシアンはギャラガへとパワーアップし，そして今またギャラガ'88となって我らがX68000に帰ってきた。時のたつのは早いものよのう。

●ギャラガ'88

おい，どうした？　元気ないじゃん。え？出演料が安かったって？　なに!?

ギャラが88(円)！

なんちて，ぱっくん，だはははははは(編：……はあ～っ)。

というわけで，電波ソフトからのアーケードゲーム移植シリーズ第何弾か忘れたけど，とにかく発売されたのでご紹介しよう。ギャラガ'88がアーケードに登場したのは，昭和88年と思った人の裏をかいて実は1988年に登場した。「ギャラガ'88」は，いうまでもないと思うが「ギャラガ」の続編だ。ゲームセンターにいる小僧に「ボク？『ギャラガ』って知ってる？」なんて聞けば10人中13人ぐらいが「知ってるサ!!」と答えるに違いない。つまり，それぐらいメジャーなゲームの**続編**なんだな。

ギャラガ'88は，実をいうとアーケードで登場したときはそれほどヒットしなかった。というのは，1988年といえば「ギャラクシーフォース」「アサルト」とか「グラディウスII」などの大作が次から次へと出た年で，ズバリ，これらの大作の陰に隠れてイマイチ注目を浴びることがなかった。なかには「ギャラガ'88ってPCエンジン専用のソフトかと思ってたよ」なんていう人もいるに違いない。

なぜか面白い

しかし，今X68000でプレイしてみると，これが面白い。どこが面白いって，そりゃゲームが面白いんですよ。意味のない拡大縮小，バカでかくてやたらに堅いボスキャラ，ここ最近のアーケードゲームは**プレイするより見ているほうが楽しい**というのが私の正直な見解。ところがこれは純粋にゲームが面白い，遊べるゲームなんですな。

まず，スタートすると，編隊を形成すべく画面外からリーダーを先頭に飛来してくる。このとき，編隊を組ませる前に全部撃破する快感といったら……。そうそう，このリーダー（黄色くて少し太ったやつね）がなかなかコシャクなやつで，人を馬鹿にしたように画面中央でクルクルと回って高速で離脱していく。こいつを仕留めたあとに出てくる「700」という数字。これを見たいがためにジョイスティックのトリガを押すスピードが速くなるわけさ。編隊を組まれてしまってからは，ギャラガたちのトリッキーな動きに騙されないように神経をピリピリ。ちぃ，こんなに緊張したのは幼稚園の入園式以来だぜ，ベイビー，たいしたやつだ，おめーってやつはよぉ。

ボーナスステージ

ギャラガを語るうえで絶対に忘れてならないのがボーナスステージ。通常ステージでのギャラガたちの動きもかなりトリッキーだが，ボーナスステージではそれにも増してトリッキーだ。しかし，相手の動きが意表を突くものであればあるほど，100パーセントを狙いたくなるというのが人情。ここまで来るかと思えば手前で引き返して去っていったり，全滅させたと思いきや生き残りのギャラガがスーッと画面の外に出ていったり。そんなときの悔しさといったら……。ちぃ，こんなに悔しい思いをしたのは高校受験以来だぜ，ベイビー。

ところで，このボーナスステージで1発も撃たずにギャラガたちのダンスをボーッと見ていると，シークレットボーナスポイントが貰える。通常ステージでのバトルで疲れた指と手を，ギャラガたちのダンスを見ながら癒すというのもいいかもしんない。

移植の出来は？

いやぁ，文句のない出来栄え。PCエンジンではカットされていた面スタート時の「おにょにょにょにょ」と叫ぶボスギャラガのアイキャッチシーンもちゃんとあるし。そういえば，最近電波ソフトはMIDIに対応しなくなりましたね。どうしたんだろ。あと，「ギャラガ'88」ってアーケードでは縦画面のゲームだったんだよね。「ドラゴンスピリット」を移植したときのように縦画面モードも付けてほしかったな。

X68000用　5"2HD版2枚組　8,200円(税別)
電波新聞社　☎03(445)6111

まとめ

X68000版「ギャラガ'88」には，例によって電波アレンジモードが付いています。ボーナスステージで，なんと，なつかしのナムコキャラクターたちが音楽にのって踊ってしまうのです。ミューキーズやニャームコたちが華麗に踊っているのを撃ち落とすなんて可愛そうで俺にはできないぜ。が，その優しさが命取りになるってことだけは忘れるな，ベイビー。

グラフィック	8
音楽	8
ゲーム性	10
お買い得度	7
何回も楽しめるぜ度	7
エンディングの感動度	9
次回作に寄せる期待度	10

自機は左右にしか動かないってとこがミソ

踊る踊る，回る回るってか

●クォース

前代未聞のシューティングパズル

Nishikawa Zenji
西川 善司

ファミコンやゲームボーイなどでも発売されている一風変わったパズルゲーム，クォースがX68000にも登場。アーケード版とそっくりのデキ，友達と２人で遊びたいゲームだ。

去年，1989年は，まさに「テトリス」をはじめとしてパズルゲーム大売り出しの年でした。その昔のニンジャブームのときには，各ゲームメーカーが揃いも揃って忍者ものを作ったように，去年（正確には去年から今年にかけて）は「テトリス」に追いつけ追い越せとばかりに，数え切れないぐらいの量のパズルゲームが発売されています。

さて，と。まだ「テトリス」ブームの余韻が残っている現在，この「クォース」を見て連想するゲームをひとついいなさい，と道行く人に聞いたとしたら……。悲しいかな，10人中９人はおそらく「テトリス」というでしょう。残りの１人は，きっと僕みたいに受けを狙って「クレージークライマー」とか「ファイナルラップ」とかいうひねくれ者だろうな。まあ，きっとパズルゲームのほとんどはこういわれるのかもしれないけど（なんてったって普通の人が知っているゲームって，「ドラクエ」と「スーパーマリオ」と「テトリス」ぐらいだもんな）。

この「クォース」，シューティングゲーム的な要素を取り入れたのは「お，頑張ってるな」と思わせるのですが，ブロックの形を揃えて消す，という点がどことなく「テトリス」を思い起こさせるのかもしれません。ま，それはおいといてさっさと紹介していきませう。

移殖の出来は？

さて，「クォース」はパッケージに「アーケード版本格移植!!」（最後の「!!」がポイント）と銘打っているようにアーケード版そっくりです。たしかアーケード版も横画面だったのでまさに完全移植，いやぁ，文句のつけようがないですな。ははは。多少アーケードのほうが解像度がよかったような気もしますが，まあ，そのへんは目をつむりましょう。音楽は原曲を聞いたことがないのでよくわかりませんが，たぶんアーケードそっくりでしょう。

クォースのルール

ルールは簡単。図のように迫り来るブロックに対して，自機の放つブロック弾で隙間を埋めて四角形をつくってやると，あら不思議，ブロックが消えてなくなり，消したブロックの数×10点が得点となります。また，まとめて消すと大量の得点が得られます。このようにして，ブロックを消しながらどんどん先へ進んでいきます。先へ進めば進むほどブロックの落下速度が速くなってきます（こりゃ，スリルだぜ）。しかも，はじめ，１面目ではなんの変哲もない緑色のブロックですが，面が進むとなんと，なんと，ブロックの，ブロックの…はあはあ…**色が変わっていくのです**。す，すごいっ。次の面はいったい何色に変わるのだろう，と手に汗握るゲーム展開を繰り広げてくれます。途中に出てくる銀色のブロックを消すと，画面上のブロックすべてが消えてくれます。で，金色のブロックがそのステージのボス（？）。これを消すと１ステージクリアです。ちなみにブロックに侵略されるとゲームオーバーです。

図１

打って，打って，消して，ああ忙しいぞこりゃ

3バリエーションの2Pプレイ

「クォース」には３種類の2Pプレイ（２人同時プレイ）モードがあります。ひとつはご存じ「対戦プレイ」。自分の消したブロックが相手側に移るというやつです。これを友達とやると友達を失いそうだなぁ。ははは（もっとも「クォース」で断たれた友情の絆ってのも情けないけど）。２つ目は「同時プレイ」。ただ単に２人並んで遊ぶだけ。３つ目が「協力プレイ」で，ひとつのブロックなんかを２人で協力して消す，なんてことが可能。対戦で失った友達とすれば仲直りっ（もっとも「クォース」で復活した友情の絆ってのも情けないけど）。

ゲームセンターでハマッタ人や，パズルファンの人なんかにはオススメかも。

X68000用	5"2HD版	6,800円(税別)
コナミ		☎03(262)9110

まとめ

「クオース」，やり始めてみると結構，面白い。どれくらい面白いかって？　弱ったな。じゃぁ「スパイ大作戦」とか「ウォーク・ワン」くらい面白いぞ，といっておこう。やっぱし，対戦がいちばん面白いかな。そうそう，縁を切りたい友達がいるのなら「対戦」よりも「協力」プレイが効果的だぞ。わざと相手の足を引っ張るのだ。「あっ，ごめーん」とかいって四角形になりそうなブロックに対して２，３発多く撃ってやるのだ。いいぞー。ごっくん。

グラフィック	6
サウンド	5
ゲーム性	5
テトリスに似てるぞ度	8
グラディウス２か３を作ってほしい度	10

●Communication PRO-68K ver2.0

シャープの通信ソフトがバージョンアップ

Yoshida Kouichi
吉田 幸一

パソコン通信ファンにはうれしいお知らせ。あのCommunication PRO-68Kがバージョンアップした。初心者でも使いやすいオーソドックスなツール。ペーパーホワイト画面で通信ができるよ〜ん。MNPにも対応している。

X68000用　5"2HD版　19,800円（税別）
シャープ　☎03(260)1161

昨日，メインで遊んでる草の根BBSのオフラインミーティング[1]とかゆーものに行ってきた。ヘンなやつがいっぱいいて面白かった。儲かりまんなあ，NTT。

Communication PRO-68K ver1.0，および私がCommunication PRO-68Kを使っているわけ

従来のCommunication PRO-68Kの重大な欠点。それはASK-68K ver.2.0に対応していないのか変換候補の表示がうまくなかったのだ。プログラムにパッチを当てるPDS[2]を入手し対応せざるをえなかった。また，このMNP時代に不可欠な“ハードウェアフロー制御”も対応していなかった。

しかし，それでも私はCommunication PRO-68Kを使っている。白地に黒の画面と80×25行しか使わない余裕のある画面構成が好きだからだ。たーみのるやMuterm[3]では画面全部がターミナルだし，それではX68000が端末専用機になった気がするのだ。せっかくだから，そんなのいやじゃん。

ver2.0への変更点，およびCommunication PRO-68Kの基礎と，清く正しい(!?)逆スクロール

Communication PRO-68Kは写真のように立ち上がる。真ん中のホワイトボードみたいのがターミナル画面。黒地に白のモードもある。**このモードだと，ESCシーケンスで色を出せたりする**（ANSIターミナルモード[4]）。ここはパソコン通信では一般的な**80×25**行。で，80×25行ということは上下7行と横16桁が余る。そこをどう使うかが**X68000のおいしさ**をどう出すかにつながる。

最上行はファンクションキー表示。最下行はかな漢字変換で下から2番目は行編集。

右の余った桁は各種情報表示に使われる。まず通信状態。送信可能かどうか，モデムからキャリアはきてるかどうか，逆スクロールモードのとき何行逆スクロールしているかが出る。で，注意すべきはキャリア検出。モデムのほうが，**キャリアを端末（つまりパソコン）へ返すモードと常にキャリアONの信号を出すモード**を持っているのだ。たとえばAIWAのMNPモデム[5]だと，デフォルトでCD信号がONになっていて，**キャリアがきてなくても「キャリアあり」表示**になってしまってなんか気持ち悪い。**モデムの設定は重要である（その1）。**

その下は伝送データ。受信バイト数やら受信行数やらそういったものが返される。

さらに，漢字コードのモード（シフトJISがMS漢字と表示されるのが少々気に入らないが[6]），カレントドライブの空き容量，現在時刻，接続時間が表示される。

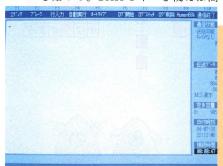
が，ここで欠点。X68000のソフトには珍しく，マウスに対応していない。**対応してもバチは当たらない**と思うのだがなあ。

さて，ファンクションキーを押して通信を始める。ファンクションキーは，エディタ（後述），ブレーク行入力，自動実行（詳しくは後述），オートタイプ（アップロードのこと），ログ開始/終了（ダウンロードのこと），ログスイッチ（ダウンロードの最中にそのON/OFFができる），ログ削除，チャイルドプロセス，終了，の10個。

さらにSHIFTキーを押すと，環境設定（詳しくは後述），受信データ同時印刷，受信データの16進表示，画面の色（ホワイトボードか黒地に白か），画面の行数（20/25），プロトコル送信，プロトコル受信，CPRO1.Uの実行，CPRO2.Uの実行，回線切断，である。

変更点は，プロトコル送信/受信とCPRO1.U/CPRO2.Uの2つ。プロトコル転送[7]については，旧バージョンではXMODEMとTransIt2だけだったのに対し，ver2.0では**XMODEMに3種類，YMODEMが新しく4つ**追加され，従来からあったTransIt2も加えて**8つ**になった。ほかにZMODEMとかもあるが，この8つがあればたいてい間に合う。

で，プロトコル転送をひとまとめにしたおかげで余った2つのファンクションキーには名前固定の自動実行ファイルが書けるようになった。これは結構便利で，**システムにはCPRO1.Uとして，MNPモデムのMNP設定用コマンド列**が登録されている。**MNPモデムには実はデフォルトでノーマルモードになっているものがあり，**その場合は自分で設定する必要がある。**モデムの設定は重要だ（その2）。**

そうだ，逆スクロール機能を忘れてはいけない。立ち上げ時の設定でメモリの許す限り（設定した値を上限として）逆スクロールバッファを持っているのだ。通信時に↑キーを押せばあらあらと，画面が逆スクロールを始める。SHIFTキーを使えば高

起動直後の画面

設定変更もこんなにわかりやすい

速スクロール(**本当に高速！**)してくれる。行単位で範囲指定してファイルにセーブしたりエディタに転送できるので，ダウンロードし忘れたり見逃したメッセージを確認したいときもOK。でも，PROと名乗るからには，任意の位置から文字列単位で範囲指定とか，**マウスでやりたい**とか思うぞ。

通信に欠かせない日本語入力だが，インライン変換モードだとうまくないので，ちゃんと変換行での入力にして使うように。

では，「詳しくは後述」シリーズ。

・エディタ

ターミナル画面の上8行にエディタウィンドウが開く。が，**既存のファイルへのオーバーライトができない**（ファイルへの追加になってしまう）などの問題はある。これはつまり，**ちょっとした送信用ファイルを編集**したり，逆スクロールバッファから転送したファイルを**ちょこちょこと編集してセーブ**するのにしか使わないのだ。たいてい，**チャイルドプロセスでED.Xなりなんなりを立ち上げる**ことになる。

・環境設定

MNPモデムに対応するために，CTS/RTS制御[8]が加わった。これで，CTS/RTSというハードウェア信号線を使ったフロー制御も可能になった。で，MNPするときには，モデム側もいろいろ設定したほうがいい。Communication PRO-68KでCTS/RTSをONにしてもモデム側がフロー制御OFFになっていてはなんにもならないからね。**モデムの設定は重要だ（その3）**。MNPについては囲みを参照してほしいけれども，**モデムのマニュアルは概して読みにくく，専門家以外にはわからないように書いてある**ので，注意が必要だ。

・自動実行プログラム，つまりはマクロ

通信ソフトを測るときのひとつのバロメータが自動実行プログラムの機能。命令数は42個と一般的。ver2.0では4つの追加関数と，5つの拡張された関数がある。

自動実行ファイルは**拡張子に“.U”**がつくから，ウィンドウからカーソルキーで，てててーっと選べばいい。このファイルは，システムについてくるのをコピーしてエディタで書き換えるのが面倒がなくて得策。

Communication PRO-68Kのおいしさと使い勝手と明日への扉

Communication PRO-68Kは初心者用としては十分速く，とっつきやすい。あとはオートログオンファイル自動作成機能とヘルプ機能，ホスト管理機能くらいだ。

通信ソフトというのは，「これだ！」という**決定打が(PC-9800の世界にも)なくて**，各社あの手この手で試行錯誤している。たとえば登録ホストをホスト名で管理したり（**閉じたデータベース**）とか，通信中に呼び出せる**IDとハンドルネームのデータベース**とかだ。そういった点ではこのソフトはひどくオーソドックスだ。だから使いやすいともいえて，**私は使っている**のだ。

だが，ホストの電話番号や名称，プロトコルやIDは立派なデータベースだ。たいていの通信ソフトはその世界で閉じている。**あまりにも前時代的**だ。私の知る限り，ホストのデータも住所録も別け隔てなくひとつのデータベースとしてとらえていたソフトはひとつだけ。それは，MZ-2500テレフォンソフト（笑）。

1) こういった集まりには珍しく女の子が多かった（11人中4人！）。ラッキー。
2) Nifty-Serveで見つけた。
3) サンデーネットが出どころのフリーウェア。通信ソフトの場合，PDSのほうが概してバグが少ない。いろんな人が使うのと，まめにバージョンアップするからだ。
4) ANSIというのは，アメリカ国内規格協会，つまりアメリカでいうJISみたいなもの。そいつが国際機関になるとISOになる。ANSIの前身がASAである。ほら，フィルムの感度をASA100とかいっていたのが最近ではISO100になったでしょ。あれはアメリカ国内の規格が国際的になったひとつの証だ。
5) ちなみに，私が使っているのはAIWAのPV24-MNP5。
6) MS漢字のMSはMS-DOSのMSである。
7) 無手順垂れ流しではバイナリファイルを送れないし，エラーチェックもできない。そこで，バイナリファイル転送用のプロトコルがいろいろとできたのである。
8) MNPの囲み参照

MNPとは

最近よく聞かれるMNP。これはマイクロコム・ネットワーク・プロトコルの略だ。

何かっていうと，エラーフリープロトコルの一種だ。パソコン通信というのは無手順非同期式，別名無手順**垂れ流し**式なのである。これだと通信速度を速くできない（文字化けやら文字落ちが出ちゃうのだ）とか，文字化けに対する対処ができないという欠点があった。そこで，アメリカのMicrocom社は考えたわけだ。モデムとパソコンの間は無手順垂れ流しのままで，モデム間を同期式にしてやれば，モデムを換えるだけでエラーフリーにできるぞ，と。

そんでもって，MNPプロトコルはできた。正確にはMNPクラス4がそうだ。2400bpsでモデム間だけ同期通信をする。そうすると，文字化けがあったりしたときは**モデム間で送り直しなどの訂正ができる**ので，ユーザーから見たらエラーフリーになるのだ。同期通信には今までみたいなスタートビットやストップビットではなく，一定の大きさのパケットごとに行う。で，パケットが一杯にならないときは一定時間入力がないと相手にデータを送るようになっている。そのために画面表示が**ときどき止まるような感じ**になるのだ。

てことは，パソコンから見るとスタートビットとストップビットを1ビット加えた1文字10ビットなわけであるが，MNPモデム同士はパケットを使うのでスタートビットやストップビットはいらない。よって1文字当たりの平均をとると，10ビット以下。そこで**モデム－パソコン間の通信速度を2400bpsより速く**してやれば，2400bpsよりちょっと速い通信が可能になるのだ。

MNPクラス5になると，モデム間の通信で**データ圧縮**をしてくれてしまうので，普通にやっているぶんには倍の速度（見かけ上4800bps相当の速度）が得られてしまう（らしい）のだ。

というわけで，モデム－パソコン間の通信速度が4800bps以上でないと意味がない。モデムが2400bpsだからといって，律儀に2400bpsでソフトを使うやつはタコなのである。MNPモデムには**モデム間の通信速度とモデム－パソコン間の通信速度が違ってても構わないよコマンドがあるのだ**。**モデムの設定は重要である（その4）**。ちなみに，その1～3は本文ね。

パソコン側からはモデム－パソコン間の速度しか設定できない。ではモデム間はどうしているかというと，モデムがよきにはからって，**相手のモデムの都合にあわせて最適の速度にしてくれる**のである。相手がMNPでなかったら，ノーマルモードで接続してくれるのだ。ただし，モデムがそういった設定になっていれば(AT¥N3)の話。**モデムの設定は重要である（その5）**。

そんでもって，私のように**モデム－パソコン間を9600bps**にするといくらなんでも速すぎる。そうなると，モデムのバッファが溢れたとき，パソコンに「ちょっと待った」といってやらねばならない。それがフロー制御であり，ソフトでやるのをXON/OFF，ハードでやるのをRTS/CTS制御という。

公衆回線を対象にしたMNPのクラスは，7,9,10とまだまだ先がある。通信の世界もハードウェア（だけ）は進んでいるのだ。

というわけで，MNPというのは，**モデム同士でいろいろとよきにはからってくれる**規格だと思えばよろしい。

ちなみに，**MNPとXMODEMを同時に使うのはよしたほうがいい**。なぜなら，MNPとXMODEMの両方でエラーチェックをやっているので無駄であり，そのぶん遅くなるからだ。

MNPしていればエラーフリーなので，テキストファイル（ISHファイルを含む）だったら，そのまま転送すればいい。以上，MNPの解説であった。ふう。真面目なことを書くと疲れる。

AFTER REVIEW

AFTER REVIEW

今月は大航海時代，プロミストランド，ウルティマV，SX-WINDOWの4つを取り上げてみました。どれもじっくり腰を据えて取り組みたいものばかり。時間があるうちに試してみてはいかが？

大航海時代

▶賭博のトランプが結構ハマるぜ！

宮城県・和田　慎一（18）

▶これは，まるで“世界ガッチリ買いまショー”だという話もある。

石川県・須田　正幸（24）

▶なかなか史実どおりで，地理の勉強になると思う理系浪人生の夏。

千葉県・安武　重人（19）

▶基本的にこのゲームはデータとの戦いである。どこにどんな港があって，そこには何が売っていて，物価がいくらで……というのを全部把握しようとするとかなり疲れる。三国志IIなどでは別に出している本のほうにいろいろ載せていたが，ゲーム本体にだって安くないお金を出しているんだから，参考資料として何か付けてほしいと思った。

愛知県・勝川　俊司（21）

▶なんといってもSLGとRPGをミックスしたような感じがいい。

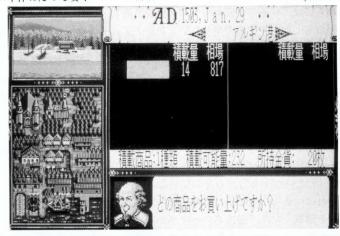
静岡県・青島　一高（22）

▶X1turboにひさしぶりに出た遊べるシミュレーションゲームなので。

栃木県・加藤　育生（17）

▶ゲームそのものはわりと面白く，地図帳とニラメッコしながらやるとなかなか遊べる（地理や経済の勉強にもなるし）。たとえジブラルタル海峡を渡るのに四苦八苦しても，がっぽり大儲けしたときの気分には代えられない。特に貯金するのが好きな人にはおすすめ。ところで，私はまだイスタンブールに入れてもらえないのだ。誰かイスラムと仲よくする方法を教えてー。　（浦川博之）

光栄といえばシミュレーション。しかし，これはいつもとちょっと違う。RÉKOEITION GAME第2弾だから，ロールプレイング的要素が盛り込まれている。だから，あのテのゲームが好きじゃない人でもとっつきやすいんじゃないかな。

X1turbo用　5″2HD版4枚組　9,800円（税別）
光栄　☎044(61)6861

発売中のソフト

★トンネルズ＆トロールズ

ここカザン帝国では，魔術師カザンによる統治の後，邪悪な死の女王レロトラーが即位し，以来激烈な戦闘が続いていた。そして今，続発する不思議な出来事に，誰もが不安を感じていた。そして皆が魔術師カザンの再来を願っていた……。

マニアに絶大なる支持を得ているテーブルトークのパソコンゲーム化。テーブルトークのシステムを用いて作られたRPGで，豊富なテキストとアドベンチャーのようなストーリー展開がウリだ。綿密な世界考証と豊富なアイテムは，本格派と呼ぶにふさわしいデキだぞ。

X68000用　5″2HD版3枚組　9,800円
スタークラフト　☎03(988)2988

★提督の決断

太平洋戦争を題材にしたシミュレーション。日本軍，連合国軍のいずれかの艦隊指令長官となり，太平洋の制海・制空権を握るために敵基地を攻略していく。プレイヤー自ら艦隊を率いて航行し，途中艦隊戦なども行いながら，敵基地を空襲，艦隊砲撃，上陸などで陥落させる。作戦の合間には国防体制の充実なども図らなければならない。太平洋の戦局は君の腕にかかっている。

9編に及ぶマルチシナリオ方式を採用。2人までの同時プレイもできるので，友達と楽しむのもいいかも。

X68000用　5″2HD版2枚組　14,800円
光栄　☎044(61)6861

★D-Again

ブロンウィンファン待望のシリーズ第5弾。戦闘シーンを配した，もはやおなじみのスタイルのアドベンチャーゲームだ。今回のブロンウィンの任務は特務機構長官護衛。しかも観光客を装って任務を遂行しなければならない。これに絡むWWWFのアングラ・マーケットの崩壊とダルジィの復活。そして出現する強棲獣曝兵器G-R。

これで第4のユニットシリーズ第1部WWWF編が完結するというからファンの人は見逃せない。

X68000用　5″2HD版4枚組　8,800円
データウエスト　☎06(968)1236

★エメラルド伝説

宇宙の果てに存在する惑星「セラ」。この星はエメラルドに収められた5つの結晶体の不思議な力に守られていた。突如現れた破壊神ソドムからセラを守るため，ひとりの少女が立ち上がる。変化に富んだ景色をバックに繰り広げられるRPG。

X68000用　5″2HD版　7,800円
システムハウスオー　☎075(502)2972

新作情報

★雀豪2

ビクター音産の麻雀ソフト「雀豪」がバージョ

プロミストランド

▶今はこれしかやってないから。

奈良県・渡辺　智一（20）

▶今までのポピュラスに加えていろいろな種類ができてバリエーションアップしたので。

京都府・和田　一博（23）

▶キャラがかわいい。

滋賀県・三橋　和広（22）

▶ひさびさに面白かった。

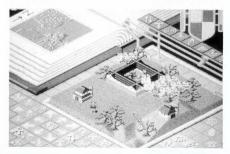
山形県・荒井　藤博（30）

▶もこもこむくむく，やめられまへん。

京都府・可児　典明（17）

▶キャラクターがとてもいい。江戸時代編がとくにいい。　神奈川県・泉田　好介（20）

ポピュラスのシナリオデータとあって人気も上々。なかでもオリジナルの江戸時代編が好評のようです。苦労した甲斐がありましたね，イマジニアさん。

X68000用　5″2HD版　4,800円（税別）
イマジニア　☎03(343)8911

ウルティマV

▶タウンガードの横をすりぬけて街へ入るあのスリルがたまらない。

埼玉県・高橋　一毅（19）

▶ロールプレイングの帝王である。

大阪府・康　正和（17）

▶前からやってみたかったから。

香川県・形幅　哲也（17）

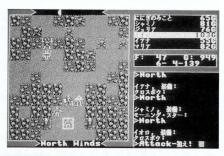
▶一生かけても解けそうにないから。

福岡県・馬場　広昭（21）

▶面白いとしかいいようがない。

埼玉県・島田　哲男（17）

Ⅳと比べると操作性がよくなったこのゲーム，英語の勉強と根気を養うのにはもってこい。さすが5作も出しているだけあって，話の作りは天下一品。解き終わるのはきっとまだまだ先だろうけど，終わったときの満足感はひとしおでしょう。

X68000用　5″2HD版2枚組　9,800円（税別）
ポニーキャニオン　☎03(221)3161

SX-WINDOW

▶これからどのようにアプリケーションが発展するのか，楽しみ。6,800円は超激安！

東京都・瓶子　卓也（33）

▶1Mなのですぐ赤旗が立ってしまう。電源を切る時の絵が気に入っているので，それでもいいのです。　兵庫県・木村　健二(25)

▶あれで6,800円はうますぎる！　欲をいえばもっとアプリが入っているとうれしい。自分でもアプリを作りたいので，今後に期待します。　静岡県・野村　一洋（24）

▶Macのウィンドウシステムよりセンスがよい。X68000の基本システムとなると思う。

福岡県・國藤　恭正（38）

発売した途端にすごい反響。皆さん気に入ってくれたようですね。これからのX68000を担うウィンドウシステムといえるでしょう。

X68000用　5″2HD版　6,800円（税別）
シャープ　☎03(260)1161

ンアップした。思考ルーチンが強化され，「強知能版」と銘うっての登場。もちろん今までどおり，自己成長型サンプリング機能と，推論型人工知能の搭載によって，打っていくたびに次第に個性を身につけ，さらに強くなってプレイヤーに挑戦してくる。麻雀そのものを楽しみたい硬派な人，もしくは友達に負けっぱなしだから，腕を磨きたいって人にお薦め。

X68000用　5″2HD版2枚組　9,800円
ビクター音楽産業　☎03(423)7901

★PIPYAN

LIFRAIMに続く，M.N.M.Softwareのパズルゲーム。

大規模なパイプ工場に突然隕石が落下，パイプに磁力が発生するというアクシデントに見舞われ，工場は壊滅状態。残された手立てはパイプをリング状につなぎ合わせること。工場の中を走り回って，散らばっているパイプをリング状につなぎ合わせ工場を救え！　制限時間もあってなかなかシビアなゲーム。面セレクトが可能なノーマルモードと，自分の行動が記録されてあとから研究できるエントリーモードがある。お好きなほうからどうぞ。

X68000用　5″2HD版　6,200円
ブラザー工業　☎052(824)2493

★FSSシナリオVol.1ペルセウスの冒険

FSS（ファンタジーサーガシステム）は横スクロールのRPG。ストーリーは，ギリシャ神話の“ペルセウスの冒険”をモチーフにしている。母ダナエーをポリュディクテース王から救うため，ゴルゴン討伐の旅に出るペルセウス。その前途に待ち受けるものは何か？

同社のアルガーナによく似たタイプで，とっつきやすく気軽に遊べるゲームだ。

X68000用　5″2HD版2枚組　6,800円
M.N.M.Software　☎0423(60)3084

★Vessel

M.N.M.Software開発の，日本ではちょっと珍しい本格テキストアドベンチャーが登場。今にも難破しようとしている船を舞台に，君は乗員・乗客をすみやかに退避させなければならない。極限状態における人間の道徳，倫理を問うストーリーが君を待っている。数々の質問にキミはどう答えるか？

500ページを超える莫大なテキスト量で，ゲームブックの感覚で楽しめるぞ。

X68000用　5″2HD版　5,000円
ブラザー工業　☎052(824)2493

★シムシティー

去年アメリカで旋風を巻き起こしたリアルタイムシミュレーションゲームがX68000に登場。1900年からスタートし，プレイヤーが市長となって町の行政を担当するというものだ。好き勝手に家を建てるだけではなく，住民の要望に合わせて鉄道を敷いたりスタジアムを建てたりなどして，住みよい町づくりを進めていかなければならない。

災害，公害，交通，税率などの幾多の困難を乗り越えて，目指すは人口25万の超巨大都市，メガロポリスだ。

X68000用　5″2HD版　9,800円
イマジニア　☎03(343)8911

★アクシス

あの，ファイナルゾーンのハワード・ボウイ大尉が帰ってくるぞ。このアクシスは，人型機動兵器を操作するアクションゲーム。X68000専用仕様で，斜め上からの視点により，立体的な画面をかもし出しているのがウリだ。スクロール速度も可変というから，気持ちいいアクションが楽しめそう。キャラクターデザインに美樹本晴彦を迎えている。

X68000用　5″2HD版　8,800円
ウルフ・チーム　☎03(5273)4795

★ぴくせる君

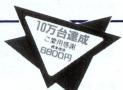
M.N.M.Softwareのスプライトパターン開発ツールが市販されることになった。

キャラクタの大きさは最大128×128ドット，65536色中16色まで使用できる。回転，反転，パレット調整機能など必要なものはひと通りそろっている。同社開発のゲームはすべてこのツールで作っているというから，性能のほどはうかがえるよね。スプライト定義のサンプルなどを添付して4,800円という価格もうれしい。

X68000用　5″2HD版　4,800円
ブラザー工業　☎052(824)2493

新製品紹介

X68000の周辺機器&ソフト近況レポート

ビデオボードCZ-6BV1
C compiler PRO-68K ver2.0
ドローイング系グラフィックツール

先ごろ発売されたビデオボードや発売が待たれるXCのver2.0などを中心にX68000の周辺事情の近況をレポートしてみよう。

●

4年目を迎えたX68000シリーズは今年も，PROII/EXPERTII/SUPER-HDなどの新機種を加え着実な歩みを見せている。また，5月号の新製品紹介でお知らせした光磁気ディスクドライブなどの周辺機器も期待の的となっている。ここで，すでに発売になったもの，それに間もなく発売されるものについての情報をまとめておこう。全体に発売が遅れているものが多いようだが，まあ，X68000も10万台を超えたことだし，ここはひとつ長い目で待ってあげよう。

まず，本体。PROIIとEXPERTIIは順調に出荷されているが，SUPER-HDについては6月に発売となったものの生産体制が追いつかず春ごろから予約していた人は結構ヤキモキしたことだろう。その後品薄状態は徐々に改善されているようだから，いまから買おうと思う人はそれほど心配することはないはず。まずはひと安心といったところだ。

次に周辺機器関係。ビデオボードが7月10日より発売となっており，これについてはあとでもう少し詳しく紹介するが，SCSIボードと光磁気ディスクについてはとりあえず秋以降と考えておいたほうがよさそうだ。

そして，ソフト。新シリーズより標準でついてくるSX-WINDOWが単体で発売された。これまでの流れを簡単に整理すると，まずPROII/EXPERTIIのシリーズに標準でついてきたもの，SUPER-HDに同梱されているSCSI対応のもの，そして今回市販されたもの，というように3つのバージョンがある。市販バージョンではSCSI対応となったほか細かい点で若干のデバッグがなされているようで，以後はこのバージョンが各製品にも同梱されている。シャープでは初期の製品に同梱されたものについては，バージョンアップサービスを行うそうで，詳しいことは直接シャープのほうに問い合わせてほしいとのことだ。

また，バージョンアップの声を聞いて久しいXCだが，現在マニュアルなどの最終段階に入っており，9月に発売できる目処がついてるらしい。

さて，SX-WINDOWやXCは大きな目玉だが，アプリケーションのほうも，通信ソフトのCommunication PRO-68Kがver2.0となって発売されたほか，新しくドローイング系のグラフィックツールがシャープブランドで近日発売の予定。期待したい。それから，Multi Wordとかいうワープロソフトが予定されているはずなのだが，その後情報がない。どうしたんだろう。

ビデオボードCZ-6BV1

ビデオボードはコンピュータ画像をビデオ信号に変換する周辺機器だ。X68000には当初からカラーイメージユニットという外付けの周辺機器があり，これを利用すればビデオ信号を取り出すことができた。しかし，カラーイメージユニットは，ビデオ機器から映像を取り込んだり（デジタイズ機能），ビデオにコンピュータ画像を合成したスーパーインポーズ画面をビデオ信号にしたり（テロップ機能）するのが主な目的で価格も69,800円とかなり高価な機器となっている。

ところが最近は，特にビデオ編集を目的としているわけでなくとも，大型テレビでアクションゲームを楽しんだり，それをビデオに録画したり，あるいはコンピュータで長編アニメーションを作るのにビデオを利用したいといった要望が強くなってきた。そこで，登場したのが単にビデオ信号を得るだけの機能に絞ったビデオボードというわけだ。

基本的にビデオ信号に変換できるのは，15kHzの低解像度モードの場合に限られている。ゲームソフトでもアフターバーナーなど低解像度モードのものは大丈夫だがポピュラスなど高解像度モードのソフトはそのままではビデオ信号には落とせないので注意が必要だ。

ボードを接続した状態で，試しにゲームソフトのワールドコートを立ち上げてみたところ，ビデオ信号ならではのギラつきと多少の色の違いはあるものの，まあまあ質の高い画像を得ることができた。起動後ディスプレイが15kHzに切り替わったところで初めてビデオ画面に表示されたが，これはコンピュータが高解像度の場合にはビデオ出力は自動的に停止するようになっているためだ。見苦しい画面を見せないための配慮だろうか。なお，ビデオボードの動作スイッチをOFFにすると，ビデオ出力は停止し本体からの出力はそのままボードを

表1　モニタ表示とビデオ出力の状態

	ビデオボード(ON)		ビデオボード(OFF)	
	モニタ表示	ビデオ出力	モニタ表示	ビデオ出力
コンピュータモード	コンピュータ画像	コンピュータ画像	コンピュータ画像	停止
スーパーインポーズモード（専用ディスプレイテレビ）	スーパーインポーズ画面（コンピュータ画面は流れる）	コンピュータ画像	コンピュータ画像が出力され，テレビとのスーパーインポーズ画面	停止
テレビモード（専用ディスプレイテレビ）	テレビ表示	コンピュータ画像	テレビ表示	停止

表2　ビデオボード（CZ-6BV1の主な仕様）

入出力端子	アナログRGB/テレビコントロール コンポジットビデオ出力/S映像出力
使用LSI	CXA1145P（NTSCエンコーダ） CXA1215M（同期信号発生）
ビデオ出力	ビデオボード動作スイッチで禁止可能 高解像度モード時は出力を停止
電源	＋5V ＋12V(コンピュータ本体より受給)
外形寸法	150mm（W）×153mm（D）×35mm（H）
重　量	260g
動作温度	10〜35℃（保存温度－25〜55℃）

素通りする。詳細は表1をご覧いただきたい。

用途が用途だけに使い方は簡単だが，ちょっと変わっているのは接続のしかた。外部への各信号はボード端のコネクタから取るのはいうまでもないが，ボードを拡張スロットに差し込むだけでなく，写真のようにボード内から出ているRGBケーブルとテレビコントロールケーブルを本体のコネクタにもつながなくてはならない。

ここでちょっと悲しいのはケーブルやコネクタがかさばるためにスロットを2つ分占有してしまうことだ。ゲームを録画するだけの人にはいいかもしれないが，グラフィックでアニメーションをという人ならメモリの増設や数値演算ボードなども考えるはず。では，なぜこういうボードのかたちにしたかというと，やはりコストを抑えるためだろう。外部ユニットにすると電源やケースで値段が上がってしまう（実際I/Oからは電源を取っているだけのようだが）。このへんもカラーイメージユニットが買えない人のためというところだろうか。

このビデオボードには通常のコンポジットビデオ出力に加えてS映像出力も備えており，S端子のついたビデオ機器の場合はより美しい画像を楽しむことができる。また，カラーイメージユニットではスーパーインポーズ画面をビデオ信号に変換するのが基本で，単にビデオに録画するだけでもビデオ信号を外部から取る必要があったが，このビデオボードでは必要ない。21,000円と求めやすい価格となっているので，目的のはっきりしている人にはお得な買い物となるだろう。

C compiler PRO-68K ver2.0

C compiler PRO-68K（通称XC）は，X68000の開発環境の中核をなす重要なセットである。アセンブラとして，Cコンパイラとして，さらにはBASICコンパイラとしても使え，X68000のユニークで強力なハードウェアをカバーするプログラマにとって必携の開発セットといえるだろう。

今回のバージョンアップの内容は，XCの高速化に加えて，ソースコードデバッガを新たに用意，さらに，ANSI規格への準拠を進める，Human68k ver2.0で拡張されたDOSコールのサポート，各種ツールも強化，といったものだ。マニュアルもさらに増え，あまりの重さにパッケージには取っ手までつくことになったという。

セットの内容は，

・Cコンパイラ
・BASIC－Cコンバータ
・アセンブラ
・リンカ
・デバッガ
・アーカイバ

および，新たに追加された，

・ソースコードデバッガ
・ライブラリアン
・プログラム保守ユーティリティ（MAKE）

などとなっている。

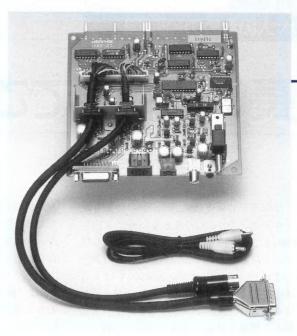
ソースコードデバッガは，ソースレベルでのデバッグを可能にするもの。フルスクリーンモード（マウス表示画面を利用）でデバッグ中にC言語のソースプログラムを行単位で実行させることができるほか，C言語で宣言された変数の参照および変更なども可能とのこと。

また，ライブラリアンは複数のオブジェクトファイルを種類ごとにまとめて管理するツール。X68000のハードをサポートする豊富なルーチン（800種以上）をライブラリとして高速にリンクできるというもの。

詳しいことは次号でお送りする予定だが，いちばん気になるのは旧バージョンのユーザーへの対応だろう。基本的には登録カードを返送しているユーザーには案内状が送られるそうだ。マニュアルも新規のものとなるので費用がどの程度になるかはまだ未定だ。

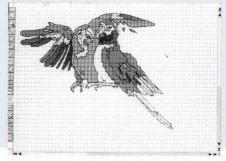
ドロー系グラフィクツール

先月の「あなたにあったグラフィックツール」の最後で，奇しくも荻窪氏が「ドローイング系のグラフィックツールがない」と指摘したばかりだが，実はこの秋，あるドローイングツールがシャープさんから発売されることになっていたのだ。

ドロー系ソフトといえば，MacintoshII用のAdobe Illustratorなどが有名だが，どうやら今回のツールはそれらを意識したものらしい。ドロー系のツールの特長は，ペイント系のツールのようにVRAM上にデータを書き込むことで画像とするのではなく，ベクトル情報からなるオブジェクトを組み合わせて絵を作るというものだ。

なにがいいかというと，一度描いた絵を再度編集し直したりすることが容易となる。また，画面上ではドット数の制約を受けるものの，プリントアウトするとずっと解像度の高いなめらかな絵を作成できるのだ。

まだ商品名が決まっていないので，ここでは仮にX-DRAWとでもしておこう。このX-DRAWは，ドローセル，ペイントセル，テキストセルという3つのセルを持っており，ドローセルとテキストセルはそれぞれベクトル情報によるオブジェクトを扱い，ペイントセルではビットマップデータを扱うことができる。これらを重ね合わせてひとつの絵を表示するわけだ。

といっても，まだピンとこないかもしれないが，詳しいことは次号で紹介する予定なのでぜひとも期待してほしい。

オブジェクトをスケルトンモードで編集

ドローセルをリアルモードにして表示

長い能書きでごめん

ついに10万台を達成したX68000。だが，そのX68000が次の一歩を踏み出すためにはどうしても欠かせない問題がある。それは「大人でも楽しめるパーソナルユース」の概念だ。単にビジネスソフトが揃うことだけでなく，ユーザーの意識も重要だぞ。

Ogikubo Kei 荻窪 圭

1987年春の発売以来，X68000の歴史も丸3年を越えた。300万台ともいわれるPC-9801でさえゲームソフトは1万本売れればヒットだといわれるこの業界で，その30分の1の10万台達成マシンX68000では1万本を越えるヒット作がスペハリ，ジェノサイド，ダンジョンマスターなどいくつも登場し，傲慢な日本の善良な市民（一応いっておくが，私が“善良な市民”というときは100％皮肉である）を驚かせた。

PC-9801も300万台のうちVM以前の買い換えられた可能性の高い古い機種や企業に大量導入した事務機器を除いたパーソナルな現場（笑）で現役している機械は100万台未満だと思われるが，それでも10万台よりは遥かに多いはずだ。ゲームユーザーのX68000にかける気合いはあなどれない。

しかし，その一方でX68000はゲームとグラフィックのマシンだという評判も定着してしまった（誰がしたのかさせたのか）。

そうなると誰が困るかというと，X68000でいろんなことをしたいと考えているユーザーと私である。「どうもゲームだのレイトレーシングだのCGAだのプログラミングだの賑やかな話ばかりだ，と。どれも好きでなきゃやれんようなもんばっかや。ワープロや表計算しよ思たら，やっぱ98にせなあかんかのぉ」という付和雷同ワールドに腰を落ち着けてしまうかもしれない。実際，そういう人も多い。

で，「大人のためのX68000」である。これはX68000がこれからメジャーになっていく過程において避けて通れない問題なのだ。でも，“大人”っていうのが問題なんだな。新しく連載やるよっていったら「アダルトソフトのコーナーですか？」なんてボケるやつは論外として（質のいいアダルトソフトがあればやりたいけど），実務ソフトを使った事務機器としてのパソコンを思い浮かべる人が大勢なのではないだろうか。

しかし，私は事務機器に成り下がったパソコンなんて大嫌いである。誰がなんといっても，嫌いである。企業なり個人商店なりが目的を持って買った，つまり必要経費として控除の対象になる機械ならともかく，せっかく個人で買ったのならそれなりに面白く使わなければ意味がない，と思うのだ。でしょでしょ。それでもって，いまのX68000は“面白いだけのパソコン”になりかねない風潮がある。それもまたよくない。

いちばん重要なもの

そこで，「大人のためのX68000」では何をするか，というと，基本はアプリケーションソフトを使ったデータの処理である。あるいはビジネスソフトで遊ぶ，ことである。データといってもグラフィックやPCMやMMLではなく，文字列や数値だ。

余談だが，理解（わか）っている人は判っているとおり，すべてのコンピュータにとってどんなデータも（プログラムなんかも含めて）等価である。単なるビットの羅列をいろいろと区別しているのは人間だ。X68000はすべてのデータは等価だということを認識させてくれる希有なパソコンである。

たとえば辞書データであるが，その膨大なファイルを“PCMファイル”とみなせば“COPY X68K_M.DIC PCM”でちゃんとノイジーな前衛音楽を披露してくれるし，画像データだとみなせばBASICのIMG_LOADコマンドや一部の人は持っているGLコマンドで画面上に意味深な模様を描き出してくれる。辞書でなくても，COMMAND.Xを鳴らしたり画像にしたりも簡単だ。すべてのファイルは等価である。余談終わり。

コンピュータを扱ううえでいちばん大切にしなければならないものは何か。それはハードウェアでもソフトウェアでもなく，溜め込んだデータである。実のところ，データさえしかるべきところでしかるべき力を発揮してくれれば，どんなマシンでも（PC-9801でもPS/55Zでも）どんなソフトでも（Lotus 1-2-3でもdBASE IIIでも）かまわないのだ。

基本コンセプトはこれである。データの共有。一度打ち込んだデータは二度と打ち直さないぞ。ってなもんだ。

あらゆる文字列化，数値化されたデータを扱うこと。大袈裟の大風呂敷のドン・キホーテないい方をすれば，これが「大人のためのX68000」の第1のテーマである。

都合のいいことにMS-DOSの5インチ2HDディスクとX68000のHuman68kのファイルには互換性がある。さらに都合のいいことに，X68000ではCSV形式というけっこう共通に使えるデータフォーマットがある。この形式で記述されたファイルはMS-DOSのいろんなソフトでも扱える。

というわけで，今月は第0回ということで，いまいったファイル互換の問題とCSV形式のデータフォーマットについての基礎知識をオマケとして記しておきたい。

もっと複雑なデータ（グラフや装飾付きのデータや計算式付きのデータなど）についてはとりあえず考えない。考え出すと，現状の悲しさを身体一杯で表現することになるだけだからだ。

ファイルの互換性について

世のMS-DOSマシンで使われているディスクは，現在のところ3.5インチと5.25インチの2DDと2HDだと考えてよい。ここで問題になるのは，2DDにも2HDにも2種類あることだ。日本のMS-DOSマシンでは2DDは640KB，2HDは1.2MBの容量になるようなフォーマットを採用しているが，IBM-PCコンパチマシンは2DDは720KB，2HDは1.44MBのフォーマットなのだ。2DDに関してはどっちのフォーマットでも読み書きできるようになっているので問題はないが，2HDに関しては（特に5インチは）互いに読み書きできるマシンは少ない。後者は2HCなどと呼ばれることもある。X68000は当然5.25インチの1.2MBタイプである。

つまり，X68000のディスクドライブで読み書きできるのは日本のMS-DOSマシンの

5インチ2HDディスクだということになる。逆も真なりで，X68000のHuman68kのディスクは5インチ2HDドライブを持ったMS-DOS上ではディレクトリを見たりファイルをエディットしたりできるわけだ。

ただし，MS-DOSではHuman68kにできるのにできない2つのファイル名制限がある。

ひとつは，英小文字のファイル名は扱えないこと。

もうひとつは，ファイル名8文字，拡張子3文字しか扱えないことである。

前者は，MS-DOSでDIRすると小文字のファイル名が見えるのに扱えないといった，歯ぎしりぎりぎりである。私も卒論の下書きをX68000で書いて，学校のPC-9801で打ち出そうとしたら，ファイル名が小文字で駄目だったという無駄な1日を過ごしたことがある。そーいえばさっきもKamikazeで作ったファイルをLotus 1-2-3で読ませようとして同じ目にあった。人間，なかなか成長しないものである。

後者はたいした問題ではない。X68000上で“OHX8ガッゴウ.TXT”というファイルを作ってMS-DOS上で読むと，“OHX8ガッコ.TXT”となるだけだからだ。

なんとかX68000とPC-9801で同じデータが扱えることがわかったわけで，めでたい。大人は寛大なので，PC-9801とだって手を組んだり利用したりするのである。

あとはシャープかサードパーティが安くて小型な3.5インチ外付けオートイジェクトドライブを出してくれればもっと完璧（間違った日本語）である。

CSV形式について

CSV形式というのはKamikazeやCARD PRO-68K，CYBERNOTE PRO-68Kなどで読み書きできるデータ記述の方法である。Stationery PRO-68Kに至っては，初めからCSV形式でデータを持っている。MS-DOSマシンでもdBASEなどで使用可能である。ちなみにCSVはComma Separated Value（カンマで区切った値）の略であり，標準形式，DIF形式などともいわれている。

具体的には，1つひとつの項目をカンマで区切り，ひとつのレコードを復帰改行「CR＋LF」で区切って記述する方法。項目が文字列であれば，ダブルクォーテーション「”」で囲む。数値はそのまま。なんていってもわかんない人にはピンとこないだろうから例を挙げておこう。

なお，KamikazeのCSV形式と，CARD PRO-68K（および CYBERNOTE PRO-68K）のCSV形式は微妙に違ってたりする。「Kamikazeは空欄（データのない項目）についてはカンマだけで処理しているが（例：”A”,,,,”B”）CYBERNOTE PRO-68Kでは空欄も””としている（例：”A”,””,””,”B”）のだ。それでも，互いにファイルをやりとりする際には困らない。

世界で1000万本普及しているというLotus 1-2-3であるが，テキスト読み込みという形でCSVに似たファイルを読むことができる。ただし，「カンマを項目の区切りとして認識しない」ため，ヌルの項目もダブルクォーテーションでくくる必要がある（””のこと）。そんなときはCARD PRO-68Kの方式が便利である。

データの共用を目指した書式には，ほかにもSYLK形式（マルチプラン用），ASCII形式（区切りなし），ASCII形式（区切りあり），などがある。X68000の場合，SYLKはKamikazeで，ASCII形式はCARD PRO-68Kでサポートしている。

SYLK形式は1つひとつの項目に対して座標が入ったり数式なども入れられるため多くの情報が表現できる。表計算に限れば有利だ。

ASCII形式の区切りなしというのは，項目を区切る区切り文字（カンマなど）がなくて，1レコードを1行として出力する方式。余計な文字が入らないので，ワープロなどに読み出すのには便利。各項目が何桁あるかがわかっていれば，区切り文字はなくてもなんとかなるものである。

ASCII形式の区切りありというのは，項目ごとに復帰改行(CR＋LF)してる形式。

これらのデータを格納する方式はあくまでも「異なるソフト間でデータをやりとりする」のが目的なのであって，たいていのソフトは独自の形式でデータを管理している。と，いうことは，「コンバート」という作業が必要になるわけである。使っているソフトのネイティブな格納形式のデータを，汎用的なテキストデータにコンバートする，あるいはその逆だ。そのことを忘れてはならない。

大人にも親切に

まあ，今回はご挨拶と予備知識だからこんなもんかな。来月からは毎回具体的なテーマを選んで話を進めよう。パソコン初心者の人も対象にするので，簡単な話から始める。若い初心者には「苦労しただけ身になるのだから，自分の道は自分で切り開こう。頭を使ってこそ人間だぁ」なんていい加減なことをいって済ますこともできるのだが（昔はそれを「Oh!MZはドラゴンである」などといって正当化していた），大人にまでそんなことは要求しないので，私はわがままな自分に鞭打って，慣れないこととは知りつつも，親切に書くつもりである。

まあ，大人なんていっても，「日本は子供社会だ」なんて断言する人がいるくらいなもんで，年齢なんて気にしないでね。大人の定義なんてどーあがいても言葉遊びにしかなんないし。まあ，大人は「謙虚」だ，とでも，いっておこうか（私がいっても説得力がないかもしらんが）。

＊

来月は第1回として，“一度打ち込んだデータは二度と手放さないぞ”の精神による住所録談話でもしてみようかと思う（陳腐なネタで悪かったわね。でも，汎用性が高いから便利でいいのさ）。

図1　さまざまなテキスト形式

CSV形式 1
”Oh!X編集部”,”03-5488-1309”,”東京都港区高輪”,”ぽよよ～ん”,,,,,”引っ越したんだよん”,
”Oh!～編集部”……

CSV形式 2（とにかく”で囲む）
”Oh!X編集部”,”03-5488-1309”,”東京都港区高輪”,”ぽよよ～ん”,””,””,””,””,”引っ越したんだよん”,””
”Oh!～編集部”……

ASCII形式 1（区切りなし）
Oh!X編集部　03-5488-1309　東京都港区高輪　ぽよよ～ん　　　　　引っ越したんだよん
Oh!～編集部……
(第1フィールド：11桁，第2フィールド：14桁……の場合)

ASCII形式 2（区切りあり）
Oh!X編集部
03-5488-1309
東京都港区高輪
ぽよよ～ん

引っ越したんだよん
Oh!～編集部
……

＊データベースソフトから変換したときはたいてい空白部分も項目の桁数だけスペースで埋められる。

清水和人流
プログラミング道場

〔その1〕

アマグラマに花道を

Shimizu Kazuto

ゲームハイテク道場，FORTH入門など数々の名作を生んだ，あの清水和人さんが帰ってきました。いや〜お帰んなさい。またまたユニークな記事をお願いしますよ。てなもんで初心者の皆さん，この講座は要チェックですよ。

アマグラマのすすめ

「人間は考える葦である」

しかし最近のコンピュータの発達により，人間の考えるべきことの多くをコンピュータが代わってくれるようになった。そのため人間は創造的な部分，コンピュータには任せきれない精神的な部分を主に担当すればよくなってきた。すなわち，計算や繰り返しや，機械的な単純作業はすべてコンピュータに任せればいいのである。それはなにも何億もするような大型コンピュータ，スーパーコンピュータに限られたことではない。我々の目の前にあるパソコンでもかなりのことを任せることができる。

無論，それができるには，いや自由にできるには少なくともBASICなどの言語を用いてプログラムを作る能力が必要である。これは誰にもたやすくできることではない。だから，すぐに使えるソフトがあらかじめ揃っていることが重視され，そうでないコンピュータは，

「ソフトなければただの箱」?

などと言われてしまうのである。

だがパソコン本来の面白さは，自分でプログラムを組み，自分の代用をさせることではないだろうか。自分の作ったプログラムのあの愛らしさを知れば，プログラムが組めてよかったときっと思える。それほどプログラミングには魅力がある。特にパソコンでのプログラミングにこそ創造性，人間らしさがあるのではないかと私は思っている。またプログラムに慣れてくると先ほどの付録ソフトの使用法もグンと応用範囲が広がってくるはずだ。それは裏でソフトがどのような処理をしているかが理解できるようになるからである。こうでなくてはせっかくの愛機が，

「ソフトあってもただの箱」

になりかねないのである。

とはいっても，自由自在に，しかも早くプログラムを組めるようになるには相当の時間がかかるだろうし，なにかとてつもない怪物に戦いを挑むような気持ちになる人もいるだろう。確かにプログラムは奥が深い。昔は売られているソフトと個人のプログラム力はさほどの差がなかったものだが，いま売られているゲームを個人で作ろうと思ったら要する時間は並大抵ではないだろう。この実力差がいまひとつやる気になれない原因となっているのであろう。

⋮

イヤア，ソレジャツマンナイナア。プロのプログラマなんてこの際無視して，我々はアマチュアのプログラマ，

「アマグラマ」

に徹するのがよいと思いませんか。キラクニイキマショオ。いやもっとおおざっぱに考えて，やれプログラミング技法とか，プログラム作法だとか，データ構造とか，アルゴリズムとか，そういう業界用語(?)はひとまず捨てて，何を作りたいか，何がしたいのか，そっちのほうに重点をおいて考えましょうよ。さああなたも今日から，

「プログラム窓際族」

になろうではありませんか。

発想は気まぐれに

ずいぶん大袈裟に始めてしまって内心「いかん！」とドキドキしているが，私もプログラムはアマチュアだから気にしないでと，さあて何をやろうかなあ……。

「何をやろうかな……」

「何をやろうかな……」

「何をやろうかな……」

そうです。いちばん大切なのは，

「何をやろうかな……」

なのです。アマチュアだから，別に売るソフトを作るわけじゃあないし，何をやってもいいじゃん，の世界なのだ。よかったね。たとえば……，

“コンピュータとおはなし”

いいですねえ。「君，寂しいかい？」「大丈夫，でもゲームばかりやってないで本当の私をさがして……」

“作曲”

そいつあいいや。いい曲ができれば大儲けできまっせ。

“方程式を解く”

……&%$%#！！！

“賢くなる人工知能”

いいけどかなり大変ですぞ。

いろいろありますねえ。まだまだこれは氷山の一角の一部のチョットでしかありません。人間の発想はとどまることを知らないのです。でも待ってくださいよ。喜んでばかりもいられませんよ。プログラムにしなければなりません。まあアマチュアだからそんなにすごいプログラムでなくてもいいのです。いま出た発想をすぐにプログラムにしてみましょう。

リスト1〜4がそれですよ。どうです，ひどいもんでしょ。これじゃあサギと呼ばれたってしょうがないですよ。工夫のかけらも見られません。プログラムコンクールがあれば最下位間違いなしです。

でもよく見ていただきたい。なんとなくこの手のプログラムのひな型のような気がしてこないでしょうか。リスト1のおしゃべりプログラムは「人間の入力に対してある法則をもって応答する」という点で対話プログラムの本質をついており，リスト2の作曲は乱数を用いて音楽を作っている点である種の作曲プログラムの骨となる部分である。またリスト3の一次方程式を解くプログラムは変数を用いて計算を行う典型的な例であり，リスト4に関しては「あなた」に関する知識をコンピュータが覚えていくのだから確かに賢くなるではないか。

エッ，プロの作るプログラムはもっと芸術的で，こんな変なプログラムはいくら組んでもダメだって？　いやあそうじゃないと思うけどなあ。どんなすごいプログラムだってその本質のところはこのようなた

わいもない発想なんですね。要はそのプログラムのメインイベントを60分3本勝負でいくか，バトルロイヤルかタッグマッチか，悪役同士かベビーフェイスを登場させるか。その発想の問題なんですよ。

その発想は実は2段階あります。

そいつは，

「何をやるか」と「どうやるか」

ですう。

そして，アマグラマにとって重要なのは「どうやるか」ではなく「何をやるか」なのです。どのようにして実現していくかはプログラムの技法にとらわれずただ思いつくままにガムシャラにいきましょう。そうすればそのうちコツがわかってきて勝手にプログラミングの実力がついていきますよ。きっと……。

いきなりド演歌

さあそれではひとつ自分を捨てて変なプログラムを作ってみよう。実は私は「変な」というところに非常に価値を感じてしまうので，どうしても「なんか変なの」的なものになってしまう。でもってある日突然，

“演歌の歌詞自動発生プログラム”

というようなものをとてつもなく，どうしようもなく作りたくなってしまったのだからしょうがない。

なぜ演歌かというと結構パターン化していて，決まり文句が多いので簡単そうだったからである。まあアイドル歌謡にしても，ロックにしても，ニューミュージックでさえもなんとなく決まり文句があり，できないことはなさそうだけどあまりよい案が浮かびそうもなかった。その点，演歌のパターン化といえばもうこれは完璧にして筋金入りである。

いや，わかってますよ。本当は，演歌だって難しいことを。ヒット曲の歌詞なんか真面目に聴くと「流石！」と唸ってしまいますよ。まあそれはそれとして認めますけどね，パロディックに考えた演歌があってもいいじゃない。そこはそれ，アマチュアということで許してもらわないと。

さて，まず作ろうと思ったからには次の段階「どうやるか」を考えなければならない。この場合演歌の歌詞のパターンをどのようにプログラムするかである。最初だからあまりすごいことを考えずに決まった長さで決まった構成で，出てくる単語だけ違えばいいじゃない，とこれまた安易な方向へ進めばよい。ここで演歌の特性が生きてくるわけである。

今回分析してみて本当にびっくらこいたのは，大部分の演歌の歌詞が七五調だったことだ。著作権法の関係上実際の歌詞を載せるのは控えたいが，読者の皆様におかれましては，ぜひとも知ってる演歌の歌詞を思い浮かべていただきたい。「オヤノチヲヒク」「モッテウマレタ」「サケハノメノメ」などの七音が圧倒的で，これに「ギキョウダイ」「サダメナラ」とか「ノムナラバ」の五音の言葉が補助をしている形がほとんどである。曲によっては，ここはわざわざ7文字や5文字になるように言葉を変えたなと感じられるようなところもあるほどである。

そうか，七五調か。なら話は簡単，7文字と5文字で演歌の中にいかにも出てきそうな言葉，「ナミダゴイ」「ヒトリザケ」「ニドトアエナイ」「オンナゴコロノ」などなどイッパイ集めてつなげれば一丁上がりということになりそうですねぇ。

そんでもってつなぎ合わせの方法であるが，非常に簡単に次のようにしよう。

　　ほにゃららららが　ほにゃららで
　　ほにゃららららも　ほにゃららよ
　　ほにゃららららは　ほにゃららね
　　ほにゃららららなら　ほにゃららだ

つまり7－5，7－5，7－5，7－5という調子である。あとはこの7と5のところにそれぞれの文字数の言葉を入れていくだけである。そしてあっというまに出来てしまったのがリスト5のプログラム。

どうです，ひどいもんでしょ。発想が発想ならプログラムもプログラムだね。まあそれでも一応走らせてみましょうか。

といって走った結果が例1である。

ウーン，確かに失敗かもしれないけどこりゃ笑える。このように素人の作るプログラムはうまく動かなくても結構面白いのである。ああよかったよかった。これでも言葉を厳選すればもうちったぁましな詩が出来そうな気もするし，試作版ということでお茶を濁すのである。やっぱり単語同士の意味がチョットはつながってないとだめのようですなあ。誰も言ってくれないから自分自身で「発想はよかったよ」と誉めてあげよう。

ところでリスト5のプログラムはチョイト気をつけなければいけないところがある。それは配列の取り方である。BASICの配列は一般に添字0から始まるのである。たとえばa(5)を宣言したとすると配列a

●リスト1　おはなし

```
 10 dim str a[50]
 20 dim str b(2)[50]={"こんにちは","さようなら","きみのなは"}
 30 dim str c(2)[50]={"こんにちは","さようなら","X 6 8 K"}
 40 a=""
 50 input a
 60 if a="" then goto 110
 70 for i=0 to 2
 80     if a=b(i) then print c(i)
 90 next
100 goto 40
110 end
```

●リスト2　さっきょく

```
 10 str a$[256]
 20 m_init()
 30 for i=1 to 8:m_alloc(i,4000):next
 40 for i=1 to 8:m_assign(i,i):next
 50 for i=1 to 100
 60     if int(rnd()*7)=0 then a$=a$+"c"
 70     if int(rnd()*7)=1 then a$=a$+"d"
 80     if int(rnd()*7)=2 then a$=a$+"e"
 90     if int(rnd()*7)=3 then a$=a$+"f"
100     if int(rnd()*7)=4 then a$=a$+"g"
110     if int(rnd()*7)=5 then a$=a$+"a"
120     if int(rnd()*7)=6 then a$=a$+"b"
130 next
140 m_trk(1,a$)
150 m_play()
160 end
```

●リスト3　ほーてーしき

```
10 float a,b
20 print "ax=b"
30 input "input a:",a
40 input "input b:",b
50 if a=0 and b=0 then print "ふてい":goto 90
60 if a=0 and b<>0 then print "ふのう":goto 90
70 print "x=",b/a
80 goto 20
90 end
```

●リスト4　じんこーちのー

```
 10 dim str a(100)[50],b[50]
 20 int i,j
 30 for i=0 to 100
 40     b=""
 50     input "あなたのとくちょうは";b
 60     if b="" then goto 90
 70     a(i)=b
 80 next
 90 for j=0 to i-1
100     print "あなたは  ",a(j)
110 next
120 end
```

はa(0) からa(5) までなので，6つの箱が確保されたことになる。このリストのように宣言と同時に初期値を与えているときは（＝{データ0，データ1，……}で配列に初期値を与えることができるのだ）1番目のデーナが0番に入るので間違えないよう注意が必要である。いや，かく言う私も最初は間違えたのであった。

もうひとつ気をつけるべきことは全角文字の扱いである。普通の半角文字を使うときはいいが，全角を使用するときは文字列の長さが倍になってしまうのである。したがって文字列の長さの宣言 STR においても，7文字なら14の長さを宣言しておく必要がある。これまた私は引っ掛かってしまって恥ずかしい限りだが，次から気をつければいいのである。

反省——人間の偉大さ

失敗は発明の母，とはよく言ったもので，人間には反省するという素晴らしい機能が備わっている。これがあるから人類が，科学が進歩するのである。反省をする人間こそがより大きな仕事を成功させる可能性を持っているのである。たとえば，私は一度食べてまずかった立食いそばは二度と食べないように心掛けている。1人ひとりが気をつけることによって，日本の立食いそば業界も発展しようというものである。

このプログラムの元々の発想は，「ハイキングーバスの中のゲーム集」などと題して「書店の窓際族」実用書コーナーに並んでいるような本の中によく出てくる「創作文」のゲームからきている。A君は誰が，B君はいつ，C君はどこで，D君は何を，E君はどうした，というようにバラバラに考えた文章をつないで面白い文章ができるのを楽しむ，というあれである。

ここでは人の代わりに乱数を用いて不規則性を狙っているが，この手法は創作的なプログラムについて非常に応用がきく。リスト2の作曲もまさに乱数の応用である。ただこのように完全に乱数にしてしまうとこれは創造とは程遠いメチャクチャになってしまうので，そこにはやはりなんらかの規則性を残しておく必要がある。これがこの手のプログラムの善し悪しを決定する部分であろう。

そんなわけでリスト5を反省して，もう少し規則性を持たせる方向で考えていこう。そのために7文字，5文字のそれぞれがどこに入ってもいいと考えるのはやめにして演歌の流れにしたがって，最初のフレーズはこの単語から，次はこの単語から選ぶ，というように場所によって選べる単語が異なるようにしてみよう。

こうするとある程度演歌のストーリーを決めなければならない。たとえば次のようになる。

≪例題≫

（いり）

ホニャラホニャラの　ホニャララに

ホニャラホニャラが　ホニャラララ

（さび）

ホニャラホニャラで　ホニャラなの

（くすぐり）

ホニャラホニャラは　ホニャラララ

（きめ）

ア～アホニャラララ　ホニャラのホニャラ

これを音符にすると♪♪♪～のような感じであろうか。歌ってみるとなんとなく典型的なド演歌になってきたような気がしてきたであろう。実際の演歌の作詞家もこうやって単語をあてはめていく作業を頭の中でやってるだけなんじゃないかとさえ思えてくる（阿久悠さんごめんなさい）。そして最後のホニャラのホニャラってところがそのままその歌の題名になったりして。

冗談はともかくこれをプログラムにしたのがリスト6である。リスト5に比べてデータがずいぶん増えたが，これは各部分ごとに5つずつのデータを持っているので必然的に増えてしまったのだ。

またリスト5では一度走らせると5つの詞を出して終了してしまっていたが，今度はデータの選び方によってものすごい数のバリエーションが考えられるので，リターンキーを押すたびにひとつ詞が出るようにしておいた。もっともこのへんはプログラムの本質とまったく関係ないのでどう作ったっていいのである。いっそのことかたっぱしからプリンタに出力して，その中からいいものを探し，コンクールに送ってひと儲けなんてのはどうだろう（ダメか）。

もちろんデータに使われた単語は私自身が考えたものであるが，こうやって乱数を使って組み合わせると自分でも予想できない面白い詞が出来たりするのである。コンピュータは機械的にしか動いていないのに，出来上がる詞は意外性に富んでいるところが味わい深いではないか。

また同じようにして詩や文章，諺，俳句なども簡単に作れそうである。要するに乱数を用いた発想法のプログラムというわけである。しかしこんな考えかたは芸術を冒瀆しているようでなにかスリルを感じてしまう。たとえば乱数を用いて作曲された音楽は芸術といえるのであろうか。

そんな哲学的なことを考えながら走らせた結果が例2である。例1に比べてだいぶ意味をなして演歌らしくなってきたかなと

●リスト5　演歌1

```
   10 dim str a(9)[14]={"もってうまれた","おやのちをひく","にど
とあえない","せけんしらずの","おもいだします","できることなら","
どこかにてます","あなたこいしい","ついてゆきます","さいてみせま
す"}
   20 dim str b(9)[10]={"さだめなら","みれんばな","もういちど","
ぬくもりを","まずしさを","とおりあめ","なみだごい","やさしさを",
"さいごまで","このわたし"}
   30 dim char c1(4),c2(4)
   35 for k=1 to 5
   40 for i=1 to 4
   50      c1(i)=(rand() mod 10)
   55      for j=1 to i-1
   56           if c1(i)=c1(j) then goto 50
   57      next
   60      c2(i)=(rand() mod 10)
   65      for j=1 to i-1
   66           if c2(i)=c2(j) then goto 60
   67      next
   70 next
   80 for i=1 to 4
   90      print a(c1(i)),b(c2(i))
  100 next
  110 print "＊＊＊＊＊＊＊＊＊＊＊＊＊"
  120 next
  130 end
```

［例1］

```
run
もってうまれた　とおりあめ
にどとあえない　みれんばな
せけんしらずの　さいごまで
さいてみせます　まずしさを
＊＊＊＊＊＊＊＊＊＊＊＊＊＊
あなたこいしい　さいごまで
できることなら　まずしさを
ついてゆきます　もういちど
もってうまれた　このわたし
＊＊＊＊＊＊＊＊＊＊＊＊＊＊
にどとあえない　みれんばな
どこかにてます　このわたし
ついてゆきます　さだめなら
あなたこいしい　もういちど
＊＊＊＊＊＊＊＊＊＊＊＊＊＊
おやのちをひく　もういちど
さいてみせます　このわたし
にどとあえない　やさしさを
ついてゆきます　さいごまで
＊＊＊＊＊＊＊＊＊＊＊＊＊＊
さいてみせます　まずしさを
できることなら　さだめなら
どこかにてます　もういちど
もってうまれた　さいごまで
＊＊＊＊＊＊＊＊＊＊＊＊＊＊
Ok
```

いうところである。まあそうはいっても，実はこのデータの組み合わせを考えるのにずいぶん時間をかけたのである。嘘だと思うなら自分でデータを入れて走らせてみるとよい。かなり的はずれの歌詞が出来上がるものである（そっちのほうが面白かったりして)。まあこれは女の恨みという典型的な恨歌の類になったが，第4節のくすぐりのあたりにやや不自然さを感じるほかは意外に面白い発想があったりして，作った私も結構遊んでしまった。

ここでプログラムを組んでいるときにやっぱり引っ掛かってしまった落とし穴をご紹介しよう。あまりこういう細かいことに気を取られていると豊かな発想を妨げるので触れたくないのだが，これも世のため人のためである。

ひとつはPRINT文で表示される位置についてである。使用した文字列の中には都合でほかの文字列よりチョット長いものが含まれていたため，ずれが生じてしまう。そこである程度文字列の長さを合わせるために長いものに揃えて，短いものには空白を加えたりしている。また；（セミコロン）を用いると余計な空白が必要になってしまうので，(カンマ)を用いた。こうすると丁度よい位置に文字列の間をあけて表示される。逆に連続して表示したいところにはセミコロンを用いた。

もうひとつは次の詞を書かせるための空打ちのINPUT文であるが，これにもなんらかの入力変数と型が必要である。X-BASICでは特に変数の種類を宣言していないと怒られるので注意が必要である。ただ従来のBASICと違うということを意識しすぎて喰わず嫌いにならないでもらいたい。従来のBASICを使っていた人がこのBASICに慣れるまでに2日とかからないであろうといわれる，それほどたいした差はないのである。

念を押しておくが，このような事柄はプログラミングの本来の姿とは関係ないのであって，このプログラムの本質は,「決められたデータの中から乱数で適当な組み合わせを作り，作詞の模倣をした」ということに尽きる。そしてプログラムの技法としては乱数と配列の使い方だけが重要である。このタイプのプログラムとしてまだまだ応用できるのでありんす。

最終調整と応用

「プログラムを愛するものは最終調整を重要視する」と偉大な思想家・孔子も言っているように（ウソよ）最後の段階で挫折してしまうのは少年マンガでも禁じ手とされているほどである（ホント？）から，不満なところはその場で直しておこう。

リスト6の不満点は第4節の「くすぐり」のところであった。まあそのついでに第3節にも手を加えてもう少しましにしたのがリスト7である。全体の単語分けとデータが少し変更されている。これをワクワクして走らせると例3のようになる（実際乱数を使っていると予測ができないからドキドキさせられて結構楽しい)。

素人としては，いや乱数としてはこのへんが限界で，プロの詞の足元にも及ばない（とフォローしておこう）が，逆にプロがこのようなツールを持ったらスランプを楽に抜けられるのではないか。

どうです，作詞家の皆さん，お安くしときまっせ。普通の皆さんはこのデータを変えて，自分の選んだ言葉で遊んでみておくんなはれ。暗い歌や明るい歌，女に味方した歌と，男の歌というように，結構その人の性格が出まっせ。逆にプログラムのプロといわれるような人はこんなチンケなプログラムでは満足せずに徹底的にやるかもしれないねえ。膨大な7文字，5文字その他2，3，4文字のデータベースを作り，さらにその意味から相性度のような評価値を入れておき，人工知能まがいのシステムで本格的なものを作る人（そんな物好きな変人はいないでしょう？）も出てくるかもしれませんねえ。

私はいまそんなことに興味があるのではなく，何をどのようにプログラム化するか，という基本の素振りを行っているのである。したがってできるだけ最小限の仕様にとどめておく方針で行きたいと思っている。あとは応用次第なのである。

実は「何を」「どのように」プログラムするか，という部分が実は最も面白く，最も忘れがちなところである。だから私は，

「暇プロ」

をおすすめしよう。

「暇プロ」

とは，暇に任せてつらつらと無計画に，しかも簡単なプログラムを組んでは壊し，また改良したりして，新しい発想を磨くのである。

●リスト6　演歌2

```
    1 int a
    5 dim int c1(10)
   10 dim str a1(4)[14]={"おんなごころの","いのちをかけた","こころがわりの","ひとりぼっちの","あなたしんじた"}
   20 dim str a2(4)[10]={"かなしさに","さびしさに","くるしさに","おろかさに","ふしあわせ"}
   30 dim str a3(4)[16]={"きょうもなきます","ないてのみます  ","くいてかなしい  ","いつもわびしい  ","こころさびしい  "}
   40 dim str a4(4)[10]={"みなとまち","なみださけ","ひとりざけ","ひとりたび","むねのうち"}
   50 dim str a5(4)[14]={"あなたこいしい","あなたいとしい","きっとまちます","いとしいひとと","あなたしんじて"}
   60 dim str a6(4)[14]={"いつまでも","さいごまで","いつのひか","きょうもまた","なみださえ"}
   70 dim str a7(4)[16]={"さびしいこころは","せけんしらずの  ","おやのちをひく  ","もってうまれた  ","どこかにてます  "}
   80 dim str a8(4)[10]={"みれんばな","とおりあめ","はぐれどり","わたりどり","かれすすき"}
   90 dim str a9(4)[12]={"いつまでも","このこころ","おもいだす","ぬくもりを","もういちど"}
  100 dim str a10(4)[14]={"はこだて","せんだい","にいがた","つがる","ながさき"}
  110 dim str a11(4)[14]={"はな","ひと","おとこ","くに","やど"}
  120 for i=0 to 10
  130     c1(i)=(rand() mod 5)
  140 next
  150 print a1(c1(0)),"    ",a2(c1(1))
  160 print a3(c1(2)),a4(c1(3))
  170 print a5(c1(4)),"    ",a6(c1(5))
  180 print a7(c1(6)),a8(c1(7))
  190 print "ああ";a9(c1(8)),"    ",a10(c1(9));"の";a11(c1(10))
  195 print"＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊"
  200 input a
  210 goto 120
  220 end
```

[例2]

```
?
いのちをかけた          かなしさに
くいてかなしい          ひとりざけ
きっとまちます          なみださえ
さびしいこころは        はぐれどり
ああこのこころ          せんだいのおとこ
************************
?
ひとりぼっちの          ふしあわせ
こころさびしい          ひとりたび
あなたしんじて          いつまでも
どこかにてます          はぐれどり
ああもういちど          ながさきのくに
************************
?
いのちをかけた          かなしさに
ないてのみます          ひとりたび
あなたしんじて          きょうもまた
どこかにてます          かれすすき
ああこのこころ          つがるのおとこ
************************
?
ひとりぼっちの          かなしさに
ないてのみます          ひとりたび
あなたこいしい          なみださえ
おやのちをひく          はぐれどり
ああぬくもりを          ながさきのやど
************************
```

これは一見なんの得るところもないような気がするが，人が考えることを一緒になって考えていては結局二番煎じになってしまいかねない。最初はくだらなくても自分で考えることが大切なのだ。ひと昔，10年くらい前は，現在基本ソフトといわれているソフトはほとんどなく，またグラフィックやFM音源などのハードもなく，皆BASICでプログラムを組むのが当たり前だった。そのため，暇プロをする人が多くさまざまなソフトが投稿されていたものだが，いまはゲーム全盛なのでBASICを使える人の割合は相当低いのではないだろうか。なんとなくつまらないとは思わないだろうか。

今月のまとめ

最後にアマグラマの重要な点をまとめたりする。

1)　何をやるかがいちばん重要。
2)　次にどうやるかが重要。
3)　最初はできるだけ簡単に考えてプログラムをまず作る。
4)　暇プロでいろいろくだらないプログラムを作り発想を磨く。

そいでもって今回出てきた項目は，

1)　文字列の配列
2)　乱数による選択

ってとこだろうか。もちろん必要のないようなことはすべて省略した。これからも簡単なプログラムばかりで行こうっと。

さて本当の最後に，今回の例を見て暗い人だと思われると嫌なので，「明るいバージョン」を作ったのがリスト8で，それを走らせたのが例4である。なんだか青春やNHKやニューミュージックというものを感じさせるものとなってしまい，自分を再認識してしまう今日この頃であった。

●リスト7　演歌3

```
   1 int a
   5 dim int c1(10)
  10 dim str a1(4)[14]={"おんなごころの","いのちをかけた","こころがわりの","ひとりぼっちの","あなたしんじた"}
  20 dim str a2(4)[10]={"かなしさに","さびしさに","くるしさに","おろかさに","ふしあわせ"}
  30 dim str a3(4)[16]={"きょうもなきます","ないてのみます  ","くいてかなしい  ","いつもわびしい  ","こころさびしい  "}
  40 dim str a4(4)[10]={"みなとまち","なみだざけ","ひとりざけ","ひとりたび","むねのうち"}
  50 dim str a5(4)[14]={"あなたこいしい","あなたいとしい","きっとまちます","いとしいひとと","あなたしんじて"}
  60 dim str a6(4)[14]={"いつまでも","さいごまで","いつのひか","きょうもまた","なみださえ"}
  70 dim str a7(4)[8]={"かもめ","たばこ","はつこい","ガラス","こども"}
  80 dim str a8(4)[10]={"このこいよ","このわたし","このこころ","こいでした","おもいでを"}
  90 dim str a9(4)[12]={"いつまでも","このこころ","おもいだす","ぬくもりを","もういちど"}
 100 dim str a10(4)[14]={"はこだて","せんだい","にいがた","つがる","ながさき"}
 110 dim str a11(4)[14]={"はな","ひと","おとこ","くに","やど"}
 120 for i=0 to 10   ・
 130    c1(i)=(rand() mod 5)
 140 next
 150 print a1(c1(0)),"    ",a2(c1(1))
 160 print a3(c1(2)),a4(c1(3))
 170 print a5(c1(4)),"    ",a6(c1(5))
 180 print a7(c1(6));"みたいな    ",a8(c1(7))
 190 print "ああ";a9(c1(8)),"    ",a10(c1(9));"の";a11(c1(10))
 195 print"＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊"
 200 input a
 210 goto 120
 220 end
```

[例3]

```
?
あなたしんじた          くるしさに
くいてかなしい          みなとまち
きっとまちます          いつのひか
はつこいみたいな        おもいでを
ああぬくもりを          はこだてのくに
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊
?
おんなごころの          さびしさに
ないてのみます          みなとまち
あなたこいしい          なみださえ
かもめみたいな          こいでした
ああもういちど          つがるのくに
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊
?
ひとりぼっちの          かなしさに
こころさびしい          ひとりたび
あなたこいしい          いつのひか
たばこみたいな          このこいよ
ああおもいだす          つがるのやど
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊
?
いのちをかけた          くるしさに
いつもわびしい          みなとまち
あなたこいしい          きょうもまた
こどもみたいな          このこころ
ああこのこころ          はこだてのやど
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊
```

●リスト8　明るい歌

```
   1 int a
   5 dim int c1(10)
  10 dim str a1(4)[14]={"いつもたのしい","きよくあかるい","とてもげんきな","わかくあかるい","あまくやさしい"}
  20 dim str a2(4)[10]={"うたごえに","ひとびとに","カップルに","あのひとに","はるのひび"}
  30 dim str a3(4)[16]={"きょうもわらった","のめやうたえや  ","これはおかしい  ","いつもげんきな  ","こころうきうき  "}
  40 dim str a4(4)[10]={"ゆうえんち","クリスマス","まんざいし","おおさわぎ","わらいがお"}
  50 dim str a5(4)[14]={"おどりおどって","みんなわになり","みこしかついで","ほろよいきぶん","あしたにちよう"}
  60 dim str a6(4)[14]={"ふえたいこ","ハイキング","くみかわす","うたうたう","なきわらい"}
  70 dim str a7(4)[16]={"ごうかなこころは","ざせつしらずは  ","でたとこしょうぶ","げんきいっぱい  ","どこかぬけてる  "}
  80 dim str a8(4)[10]={"おやゆずり","いいきもち","あほうどり","たいこもち","むせきにん"}
  90 dim str a9(4)[12]={"けんこうな","おもしろい","おどろいた","おかしいね","やめられぬ"}
 100 dim str a10(4)[14]={"しあわせ","たいよう","かんぺき","とこなつ","こんじょう"}
 110 dim str a11(4)[14]={"やつ","まち","おとこ","おんな","こぞう"}
 120 for i=0 to 10
 130    c1(i)=(rand() mod 5)
 140 next
 150 print a1(c1(0)),"    ",a2(c1(1))
 160 print a3(c1(2)),a4(c1(3))
 170 print a5(c1(4)),"    ",a6(c1(5))
 180 print a7(c1(6)),a8(c1(7))
 190 print "ああ";a9(c1(8)),"    ",a10(c1(9));"の";a11(c1(10))
 195 print"＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊"
 200 input a
 210 goto 120
 220 end
```

[例4]

```
?
わかくあかるい          ひとびとに
いつもげんきな          わらいがお
あしたにちよう          ハイキング
げんきいっぱい          たいこもち
ああおどろいた          たいようのおとこ
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊
?
きよくあかるい          うたごえに
こころうきうき          まんざいし
みんなわになり          ふえたいこ
ざせつしらずは          おやゆずり
ああおかしいね          しあわせのまち
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊
?
とてもげんきな          あのひとに
こころうきうき          ゆうえんち
ほろよいきぶん          なきわらい
ざせつしらずは          いいきもち
ああやめられぬ          とこなつのまち
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊
?
わかくあかるい          あのひとに
こころうきうき          まんざいし
あしたにちよう          うたうたう
どこかぬけてる          たいこもち
ああおかしいね          しあわせのまち
＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊
```

●特集1

日本語を処理するための序章

日本語

パソコンの主要な用途として挙げられる「日本語ワードプロセッサ」。パソコンの性能を測る目安として，どれだけ優秀なワープロがあるかというのを挙げる人も多い。誰もがもっとも必要とされるアプリケーションだと信じている割に，通常，ビジネスアプリケーションとして捉えられている，それが日本語ワードプロセッサだ。
ふつうの人はそれほど文書を書かないというのは事実だし，多くの人は快適な入力より，美しい出力を求めている。では優秀なワープロとは？
日本語処理が真に私たちに必要なものであるなら，環境の改善はなににもまして必要なことだといえる。同じ処理系を使っていても，身の不幸を嘆くばかりでなにも努力していない人と地道に環境改善している人では大きな違いがある。信じよう。パソコンとは，使い込めばそれなりに応えてくれるものだと。

CONTENTS

特集1

日本語を処理するための序章

ワープロを使う前に

日本語を書くための7つの方法

Yoshida Kouichi
吉田 幸一

素晴しいワープロがあったとしよう。さあ，君はいったいなにを書くのだろうか？　数えきれないくらい世に出ている日本語ワープロはいったいどのように使われているのだろうか？　ここでは「文字による自己表現」の手法を考えたい。

「ワープロがあるんだけれど，それでなにをしたらいいのだろう」

高校生の頃，部の後輩に“詞”という名前の可愛い女の子がいた。“ことば”と読むのだそうだ。うーん，いい名前だなあ。もう1回会いたいなあ。きっと，どこかで主婦してたりするんだろうなあ。と，今月の特集が日本語だというのを聞いて，思い出してしまった。

さて，ビジネスマンは手紙を書いたり，始末書を書いたり，企画書を書いたり，その他諸々の報告書のためにワープロを使う。私は仕事としてワープロを使う。学生はレポートや卒論のためにワープロを使う。ちょっとしたチラシなどA4用紙1枚以下の文書のためにワープロを使う。

全部，実務的な道具としてのワープロである。それ以外でワープロを使っているユーザーはどのくらいいるのだろう，というのはなかなか疑問であった。ワープロって結構謎なのである。ワープロというものを見ていると“文章を書くことは誰しも自然に行う行為である”みたいな扱いをされているから。

たとえば最近の若者は文章を書かない，みたいないい方をよくされる。それは間違いである。その証拠に，パソコン通信をちょっとやってみると，くだらないことを書いた文章に山ほどお目にかかれるはずだ。やはり，機会があれば誰だって文章は書くものなのだ。

趣味のワープロとは

最初にワープロソフトを前にして，あなたはなにをしただろうか。おそらく，毒にも薬にもならない文章を打ち込んで喜んだのではなかろうか。私は，X68000を買ってワープロを立ち上げたのはいいけれど，なにも書くことがなく，近くにあった哲学書を打ち込み始めたやつを知っている。

昔，アルバイトでワープロ教室のインストラクターをやっていたとき，休憩時間にサザンの歌詞を一心不乱に打ち込んで喜んでいた女子大生の受講者がいた。

これがもう少しワープロに慣れてくると，ギターを抱えてキーボードを前にして詩を書くヤツが出てきたり，小説を書くヤツが出てきたり，意味もなく論文めいたエッセイを書くやつが出てきたりする。プリンタなんかなくても，発表する場がとりあえずなくても構わない。

私はこう考えるわけである。文章を書くことは，Z's STAFFでなんか絵を描くこととか，ギターをかき鳴らすとか，漫才をするとか，プログラムを書くとかと同じく，表現の一種なのだ。“人間は～する動物である”シリーズのひとつに，“人間は表現する動物である”っていうのを入れたいくらいだ。人間は自己を表現しようとする本能を持っている，なんて怪しげな言葉にしてもいい。

喋るのが主な表現である人もいるし，絵を描くのが表現な人もいるように，書くことも一種の表現に過ぎないのだ。

そして，ワープロというのは，表現を支援する道具なのである。

書き残すことは恥ずかしいことである

さてさて，文章を書くことは文明に冒された人間の自然な表現の一種であるということにした。その証拠に私の部屋を引っ掻き回すと，書きかけの小説やらエッセイやら詩やらを残した原稿用紙やレポート用紙がポコポコと出てくるのである。ワープロはそういった表現を援助する道具でありメディアなのだ。

なんだかんだいって，我々が日本語で(少なくとも意識のうえでは) ものを考える限り，ふつふつと湧いた意見やアイデアを残すにはまだまだ文章や詩というのは有効だ。少なくとも映像や音楽よりは曖昧でなく，新しく修得すべき技術も少ない。

ワープロをおもむろに立ち上げて，適当なファイル名で書き留める。

そうした文章は，何年かたって読み返すとたいてい読むに耐えないものだったりするが，そこで消してはいけない。手を加えてもいいが，消してはいけない。重要な“青春の汚点”，“過去の汚点”なのである。私なんて汚点どころか“過去の汚面”くらいたくさんある(汚点→汚線→汚面→汚体と次元が上がる)。なんというか，若気の至りというか，若気のアンというか，そんなもんである。

表現に対する衝動について

表現することと，なにかを創り出すことは同じ。表現したいものが自分の中にしかなかったとき，他人の真似や引用でなく自分が表現できたとき，それを創造という。

表現するという行動において重要なのは，それが自分にどうフィードバックしてくるかにつきる。表現物が他人の目に触れない場合でも，頭の中で思っているよりより文章にするほうがフィードバックは大きい。さらに，他人に読んでもらったり聞いてもらったほうがフィードバックは大きい。

小さな子供は大人や友達に相手をしてもらいたくて，あの手この手を使って表現する。結果がフィードバックされて，うまくいった方法は何度も使い，うまくいかなければ新しい手を考え出して，学んでいく。それはいくつになっても(ガキのように望んだフィードバックを得るためにすねたり駄々をこねたりはしなくなるけれども)同じことだ。自己顕示欲，なんていうといい意味にとられないことも多いが，誰でもそういった衝動はある。人に見せられない衝動なら，自分にだけわかるように表現すればいいではないか。

頭の中でわかっているように思っていることでも，頭の中では理路整然としているように見えても，いざ文章にしようとするとうまくいかない。実のところ，人は言葉だけでイメージするものではないからだ。言葉でないものも言葉にしなければならな

いからだ。それには多少の訓練は必要かもしれないが，こと日本語に限っては，誰しも素養はあるのだから，そう大変な作業ではない。

言葉を積み重ねていくことによって脳は成長していくのだ。

ワープロだとこんなにおいしい

そして，ワープロというのは手書き文字と違い，客観的なフィードバックが得られる点で秀逸である。間欠的に湧いた考えをまとめたり，無手順垂れ流しで吹き出た単語の羅列に意味を持たせるよう加工するのに最適な道具なのだ。決して紙とペンの代わりではなく，もっと積極的な支援ツールなのである。綺麗な文書を作るためではなく，言葉を書き留めるための道具なのである。プリンタはなくても構わない。

ワープロで書くことのメリットをまとめてみよう。

1) 書くことによって自分の考えや言葉を客観的に見ることができる。

2) いー加減な考えでも，それなりのものにまとめることができる。

3) 自分の考えのあさはかさを知ることができる。

4) 想いを吐き出すことによってカタルシスを得られる。

5) 自分の考えていることを知ることができる。

ほら，こんなにある。ワープロによって書かれたものを伝達に使おうとすればプリンタは必要だけれど，プリンタを使わなくてもこんなにメリットはあるのである。手書きと違って，読み返すことが簡単だからだ（そんなのは，やたら字の汚い私だけだろうか）。

文章の書き方入門

“覆水盆にかえらず”という中国の有名な故事がある。昔，中国の越に覆水という大変親孝行な青年がいた。彼は毎年お盆になると，都から田舎の母親のところへ帰っていたのだが，ある年，道中で事故があって，帰省できなくなった。突然のことで便りもできず，覆水が帰ってこないことをはかなんだ老齢の母親はショックで死んでしまったという話である。教訓として，普段孝行なばかりにちょっとした事故で親を嘆かせてしまった。適度な親不孝は必要である，と，私に有利な結論が導き出せるのであった。

もちろん，この話は嘘八百。しかし，“覆水盆にかえらず”という言葉しか知らない人であれば，信じたかもしれない。単独で絶対的な意味を持つ言葉なんてないのである。その言葉が連なって作られた文もまたしかり。

第0講「言葉にするということ」

こういう大原則がある。

“人は自分の知らないことは書けない”

さらに，こう続く。

“人は言葉として知るとは限らない”

私は，マグリットの1枚の絵を見て感動した。その絵を描写することはできる。それは映像を言葉に変換することである。その際に私は，その絵を描写するに相応しい自分の知っている言葉に置き換える。それを解釈という。自分の知っている言葉で表せないと（たいていそうだが）石になる。

なんらかの言わんとすることは，映像や音や動きや言葉やいろんなメディアでイメージされ，それを映像で表現するときはイメージのすべてを映像に変換しようと試み，文章で表現しようとするときはイメージのすべてを文章に変換しようと試みるわけだ。ああ，無謀。

映画を見て感動したとき，小説を読んで感動したとき，ゲームをやって感動したときなどなど，人はそれを表現したいと思う。

表現する手段が文章のとき，

“表現するに必要な言葉の知識”

“自分がなにに感動したか見つめる心”

が必要になる。脳よ，働け。

商業誌など不特定多数の人を対象にすると，それに，

“自分が感動した理由の分析”

や，

“主観と客観の分離”

が追加される。

どれも非常に厄介な問題である。今回はそういった厄介な問題に取り組むわけだ。

表現しようとするなにかを自分の中に持つこと，それがいちばん重要であるが，そのくらいは誰にだって，あるよね。なければ，人間として問題だと思う。ははは。

表現したり感動したりするために必要なのは，経験だ。人は自分の経験にまったく引っかからないものに感動はしないし，経験なくしては表現はできない。この経験というのは生まれてからいままで見たり聞いたり触ったりして，その人の感覚に対する入力すべてを指す。

創造するなんていっても，なんの経験も

人に読ませるということ

よくいわれるのが，学校の先生でもいう程度のことだが，

“読む人のことを考えろ”

である。これはたいていの場合無駄である。書き慣れない人は“自分とよく似た感性，嗜好を持った人しか想定できない”から。これは文章に限らず，音楽でも，マンガでもよく似たものだ。

代表的なのが比喩の使い方だ。Oh!Xだからこそ“〜はドラゴンである”とか“〜はその筋である”なんて言い回しが通用するのであって，これを違う言葉で表現しようと思うと，非常に苦労する。書き換えによって“〜はドラゴンである”という言葉自体の持つリズムをなくすと同時に，正確に表そうとすると，論文のような理詰めにならざるをえないからだ。比喩は便利である。しかし，乱用は慎むべきだ。

というわけで，想定すべき読者は以下のように考えるのが正しい。

“その文章を書くにいたった経緯やそのときの心情など，その文章にまとわりつく記憶がない状態での自分”

である。

これは結構簡単なようで難しい。書いている時点の自分にとって当たり前のことでも，記述しなければならないことというのは，多いものなのだ。かといって，いちいち全部書いていてはただのなにがいいたいかわからない駄文。

面白いことを思いついて（思いついたと思い込んで），とりあえずエッセンスだけでもメモにしておこう，と書いた落書きは，たいてい1週間もすると，なにがいいたかったのかわからなくなる。これは情報が足りなかったのである。そのときの状況や心情が思い出せてはじめて意味をなすのだ。こんな文章を書いていてはいけない。

リズムや流れを保ったまま，必要な情報を盛り込む。これは一種のテクニックを必要とする。偉大なる先達の作品を読むのがいちばんいいだろう（結局，読むことは大事）。

ないところからポンと創り出すなんて不可能なのだ。これは本当。
だから，経験は広く多いほうがいいのだ。

第1講「上手な文章を書けない理由」

誰でもいくばくかの経験はあり，言葉をある程度知っており，本を読んでいるはずだ。経験はすべて無意識のうちに蓄積される。だから誰にでも書けるはずなのが文章である。
しかし，そうは問屋が卸さない。“自分が思ったことを論理的に記述することは不可能”だからだ。それ以前の主観と客観の分離だって難しいというのに。
そこでたいていの人は論理的な記述を諦め，感覚に訴える，目に優しい文章を試みる。文学的な表現をしようというわけだ。これも失敗する。理由はいろいろとあるが，
“まともな文章を書くためのルールを知らない”
“言葉を知らない”
などによって，思ったように進まないのだ。これはプログラムを書くときに，
“アルゴリズムが記述できる”
“その言語を知っている”
という条件が揃わないと駄目なのと同じである。テキトーな言葉を連ねて，わざとわかりにくい文章にしてごまかすという手もあるが，そんなのは“エセ哲学屋”に任せておけばいい。

第2講「読みやすい文章とは」

ここで期待している人をがっかりさせよう。たいていの人は“読みやすい文章はすなわち，いい文章”だと誤解しているからだ。
正しくは，
“読みやすい文章とは，すなわち，パターン化された表現が多用され，読者の持つ価値観や経験（疑似体験を含む）から逸脱しない文章”
なのである。
赤川次郎の文章は読みやすい。しかし，名文ではない。読者が手を抜いても読めるような書き方で，その程度の内容しか書いてないからだ。読みやすいのは当たり前である。
たとえば中世の古城を表現するとき，
“中世に建てられたような旧びた古城があった”
とする。たいていの日本人は自分なりの石でできた古城をイメージする。いまの時代，中世の古城といってなにも思い浮かべられないやつはいない。しかし，それがどんな城であれ，それで終わりである。「作者のイメージする城」はどこにもない。
昔，平井和正というSF作家が，星新一に文章がすらすら読めるようになった，といわれて嘆いていた（曖昧な記憶だけど）。平井和正は比喩や修飾の多い，一見複雑な文章を書いていた。それがすらすら読めるとはどういうことか。表現のパターンが読者に読まれてしまい，一言一句追わなくても，その内容がわかってしまうようになったからである。つまり，平井節（といわれる文体）に読者が慣れてしまったのだ。
というわけで，読みやすい文章がすなわち名文とは限らないのである。そこを勘違いすると，他人の作り上げたパターンの表現を積み重ねただけの文章しか書けなくなってしまう。人気のある村上春樹の文体だって，よく読めば，アメリカのハードボイルドの文体をパクって日本風にしただけ，にしか思えない。
他人の文体をパクることはカッコいい文章には欠かせないが，パクっている自分を意識することが必要だ。
意味もなく複雑で読みにくい文章は論外だが，読みやすい文章を目指すあまりに，必要な情報まで捨ててしまってはいけない。読みやすい文章を目指すのは次の段階である。

文章に重要な流れ

文章には流れがあり，リズムがある。流れがスムーズだと，内容がわからなくても素直に読めるものである。
その究極の形態が短歌であり，俳句だ。あんなもんを解釈して喜ぶのはタコである。そこのところを高校の古典の教師はわかっていない。あれは，古代の歌謡曲なのだ。気持ちのいい言葉を並べて，なおかつ，内容がまとまっていれば勝ちなのだ。
5・7・5・7・7っていうのは，音楽でいうとロックやレゲエやジャズなどのリズムと同じであって，ときどきある字余りはシンコペーションなのだ。聞く人間は詠まれた句を聞きながら，予想をはずされたり，思わぬオチを聞かされたりして感心していたに違いないのだ。句会なんて，セッションと同じだったのだ。
そのことに気づいたのが大学に入ってからだった私は，中学・高校の6年間，古典の時間を無駄に過ごしていたことになる。うーん，くやしい。
散文でも同じようなものである。
自分が書こうとする内容を，いかにちゃんとした流れで表現するか。文章にリズムを持たせるか。それが問題だ。流れがしっかりしていれば，読者を誤魔化すこともできるし，書きながら「こんなはずじゃなかった」なんて思うこともない。
もし，どうしても頭の中で考えたような流れの文章にならないのだとしたら，頭の中で考えた流れが，スムーズでなかった，あるいは間違っていたわけで，それはそれでひとつ頭がよくなったと思って喜ぶべきである。きっと，書く行為なしでは気がつかなかったろうから。
しかし，自分で感じたような流れを保って文章を書くのは難しい。たいてい，予想だにしない方向へ筆は進み，あらぬ結論を出そうとしてしまう。特に書く行為は，人の考えを極端にしていく。書くことによるフィードバックによって，それまでに書いた内容が増幅されて次の文に反映されるからだ。
これは書くより考えるほうが数倍速いことも理由のひとつだろう。人は言語だけで考えているわけではないから，文章にするときは言葉になっていないところや飛躍した考えを補完しながら書く必要があるからである。
やばい，と思ったとき，よく使う手が“さて”などという話題転換の術や，“うーん”という擬音で誤魔化す術である。なるべくならそんな術は使わないにこしたことがない。
思考や感覚を流れのまま文章にできたら，それで第1段階は終了である。言葉の使い方がおかしかろうが，文法的に変なところがあろうが，美しい文章になっていようがなかろうが，流れがしっかりしていれば，読む者は誤魔化されてしまう。なかには祝一平氏のように誤魔化されない人間もいるが，そういった読者はとりあえず考えなくてもいい。そんなのは次の次の段階だ。
人は流れで文章を読むものである。一言一句論理的に追いかける人は（私みたいに怪しいことを書くライターの文章をチェックする不運の編集者はさておき）あまりいない。“哲学書が（良質のものであっても）読みにくいのは，事前に知識を要求することだけでなく，普段我々が読んでいる文章のとは異質の流れ・リズムの上に成り立っているから”である。翻訳文が読みにくいのも同様だ（最近では翻訳文もひとつの文体として市民権を得ているので一概にそうとはいえないが）。
とはいえ，読者に少しでも理解してもらおうと思ったら“流れが論理的に追えるもの”であることにこしたことはない。

第3講「悪文で育つと悪文しか書けない」

作文は，生まれながらにして持っている本能ではない。後天的に表現手段の一種として身につけたものである。よって，悪い環境で育つといい文章は書けない。というわけで，悪い文章を紹介しよう。我々がいかにひどい文章に囲まれているか知ってぞっとするがよい。
1) ……栗本は野上と出かける約束をしていたが，時間になっても待ち合わせ場所に来ないので，電話をしたが，出なかった。そしてマンションまで行き，声を掛けたが返事がないので中に入ると，風呂場から水の音がするのでまた声を掛けた。……

(Misty 4より。Oh!X 8月号30ページ左下の写真参照)

これは会話文なので悪文の例としてはちょっと不適当だが，こうして活字の文章に置き換えるとなにが悪いか読者諸氏にもすぐにわかるだろう。

まず，8つの行為が2つの文章に詰め込まれており，リズムがない。事象の羅列で，その事象が句点によって4つもつながっていて，“～が”と“～ので”が乱発されていて見苦しい。

実際，我々の日常会話はというと，おそらくもっともっとひどい日本語をしゃべっているに違いない。だが，仮にも第三者である読み手にイメージを再現させることを前提としたメディア（小説でもテキストアドベンチャーでも）では，ある程度を整然とした文章（たとえ非現実的でも）を用いるようにしたい。

2)　近くの森をぶらついているとき，君は不思議なジプシーのウワサを誰かがしていたのを思い出していた。

(ウルティマVより。Oh!X7月号30ページの画面写真参照)

1)ほどひどくはないが，変な文章である。“～とき,～思い出していた”というつながりが変なのだ。“～しながら,～思い出していた”か，“～とき，～思い出した”でなければおかしい。翻訳のためかと思われるが，もう少し綺麗な日本語にならなかったろうか。

書き換えの例を検討してみよう。前後の文章によってどれがいいかは異なるが，同じ内容の文を異なった文にしてみるのもいい勉強になる。

A）“君は近くの森をぶらついていると，誰かがしていた不思議なジプシーの噂を思い出した”

B）“君が，誰かがしていた不思議なジプシーの噂を思い出したのは，近くの森をぶらついているときだった”

元の文の語順では，近くの森をぶらつくことと，ジプシーのウワサとの関係が気になる。B)だとウワサを思い出すことがメインとなり，森はただその思い出した行為を補足するだけになって，文としてのおかしさはなくなる。

3)　久しぶりに体を動かしたくなった俺は後先を考えずに男どもと乱闘になった。

(アルビオンより。Oh!X 2月号27ページの画面写真参照)

主語は“俺は～”で単数だから“乱闘を始めた”くらいでないとおかしいが，“～になった”とはだいぶニュアンスが変わってしまう。実は“俺は男どもと乱闘になった”だけならそれほどおかしくは感じないだろう。意味的には複数の主語として読めるからだ。ところが，欲ばってつけた“後先を考えずに”がまずかった。これは“俺”個人の状況を説明するものでありながら，文法的には複数の行為者を主語に持つはずの“乱闘になった”という述部を修飾しているのである。より詳しく状況を説明しようとしたために陥る罠といえるだろう。

なお，“～なった”を生かすための簡単な解決法がある。試しに，“考えずに”を“考えず,”に替えて読んでほしい。

4)　スー　「きゃあぁぁぁぁぁあっ！」
　　ピクト「うわぁぁっ！」
　　スー　「ま，まぶしい……。」

(アークスIIより。Oh!X 4月号89ページ右上の写真参照)

べつに文章が悪い例ではないが，この台詞の上のグラフィックはキャラクターが抜いた剣がまぶしく光っている様を描いている。つまり，まぶしいのは絵を見れば一目瞭然のこんこんちきなのである。ウルフ・チームお得意のビジュアルシーンがこの台詞のおかげで台無しになっている。せめて，“こ，この光は……”というように絵では表せない部分を想像させるような工夫がほしいところである。

と，身近なゲームから例を挙げさせてもらった。とりあえず，画面写真から認識できるものから探しただけでこんなにあった。ゲーム画面，ソフトのマニュアル，チラシ広告，どこも「お粗末」な文章であふれている。こんな文を読んで育ったら悪文しか書けなくなるのは目に見えているぞ。

さて，一般的な問題に入ろう。

犯しやすいのは，文の構成がスパゲティプログラムのようにグチャグチャになることである。ひとつの文章が長くなりすぎると，FORに対応するNEXTがありません，てなことになる。

“焼けたアスファルトを歩き疲れた僕は，開け放たれた窓の向こうで仰向けになって転がっている猫が笑い，長い前髪に隠れた伏し目がちな顔の彼女がこの世の全てを許したかのような表情でそれとじゃれあっていた”

という,いま思いつきで書いた文章である。いやあ，中学生でも書かないようなひどい文章だが，まあ，許してくれ。この文章の構造を見てみよう。主語は“僕”である。しかし，それに対する述語がない。文が長くなる内に（書き手が忘れてしまって）述語がどっかへ行ってしまったのである。これがFORに対応するNEXTがないというヤツ。この場合,“僕”の行為の中に“猫”と“彼女”の行為がある入れ子なわけで，“僕”の行為だけが完結していない。思いつきで書いているとよくやりがちなボケだ。

こういった文章はもっと短く，いくつかの文に分けるべきである。そうならないのは，頭に思い浮かんだ順番に書いているので，書きながら思いついたり思い出したりしたことをそのまま文に付け加えるからで，推敲が必要だ。

語順を少し変えるだけで見違えるほどすっきりした文章になることは多い。

続いてよくある例が，関係詞的な構造を入れたばかりに文が複雑になるケースである。元来関係詞的な文（～するところの～，っていうやつ）というのは現代語にはなかったものだ。英文の翻訳が一般的になって以来のことで，便利屋的に使うと見苦しくなる。

次もよくある話で，倒置法の乱用である。“～であった，～だったが。”っていうやつである。

上の3つはどれも根っこは同じだ。

“思いついたものを整理せず，思いついた順番でそのまま書いてしまうので，文全体の帳尻をあわせようとして，複雑で不自然な文になってしまう”のである。

思いつくままに書くことと，感覚の流れをそのまま文章にすることとは違うのだ。

第4講「悪文を書かないために」

最初からそんなことを気にしていては文章なんて書けないわけで，書いてから直すのが基本である。そのためにはまず，

“時間をおいて読み返すクセをつけること”

が大前提であり，さらに，

“自分で書いた文章が悪文かどうかを知ること”

が重要である。特に自分で書いた文章は“その文を書いたときの状況や心境を踏まえて読んでしまう”から，言葉のあいだに隠れたニュアンスを読み取ったり，書き間違いに気づかなかったりしがちだ。

最低限，悪文かどうか，この文はおかしい気がするなあと思えるようになると次の段階である。気がついても，なかなか直せるものではない。そのためのコツというものは確かに存在する。

1) 文体の統一をチェックする。

ですます調とである調を混在させない。

2) 同じ文末を続けて使わない。

文の終わりがすべて“～だ”だったり，“～である”だったりすると，きれいでない。かの三島由紀夫だって，文末が“だった”ばかりになるのを防ぐために，意識して現在形をまじえたりしていたという。私でさえ，同じパターンを繰り返さないよう，ときどき体言止めや倒置，疑問形などをまじえるようにしている。こういったことは推敲しないと気がつかない。

3) 代名詞の使い方をチェックする。

“あきらは”という主語がいくつも続くとくどいので，“彼は”にしたり，主語を省略したりしてみる。

4) 40字以上の文は2つに分けられないか考えてみる。

ひとつの文にたくさんの主張を持たせてはいけない。私はよくわざと長い文を書いたりするけれど。

5) 同じ表現を何度も使わない。

6) 語順を変えてみる。

語順というのはリズムのためにも，簡潔な文を書くためにも重要である。

7) 接続詞を全部取ってみる。

接続詞がなくてもわかる文章を書ければ一人前だ（これは大変だぞ）。

そんなこんなで，文章というものはプログラムと一緒で，デバッグを重ねるとこなれてよいものになっていくのだ。

第5講「カッコいい文章」

いい文章と悪い文章というのは確かにあって，前述の三島由紀夫なんかが名文としてよく挙げられるけど，無理やりそう思う必要はない。文章というものは時代に応じて変化していくものであり，少なくとも我々は三島の時代とは異なった文化にいるからだ。私が現在いちばん綺麗な文章を書くと思う作家は筒井康隆だ。

私より若い世代だと，それが吉本ばななになっているかもしれない。

話し言葉が変わる限り，書き言葉もそれを無視してはいられないのである。

いまは急激に変わってきた話し言葉と書き言葉のギャップが大きくなったために，そのあいだを埋めようとする，新しい言文一致運動の時代だそうだ（科学朝日8月号，「ネオ日本語」）。

というわけで，カッコいい文章というのは，書き手の育った文化に大きく依存する。いまさら三島由紀夫の文章を勉強せよとはいわない。しかし，文章を書く目的を表現の伝達と考えれば，なるべく簡潔で十分に必要な情報を盛り込んでいるものがいい文章といえよう。

第6講「言葉は記号である」

いまさら記号論でもないけれど，言葉というのは，共同幻想の上に成り立つ記号でしかない。

だから，書き手と読み手とのあいだでその幻想が成り立っていれば，どんな言葉を使おうが知ったことではない。逆に，幻想が成り立たない言葉でいくら語られても知ったことではない。そういうものである。

古くからある，すでに意味や用法の確立された言葉があれば，その用法を逸脱してはならない，ということだ。言葉を崩すのが流行っているが（例：本章の冒頭），元の意味を知らずにそれをやっても見苦しいだけだ。

＊　＊　＊

てにをは，とか，接続詞の使い方とか，そういった実務的なテクニックを求める人にはかなり拍子抜けしたことと思うが，言葉が流動的で，誰もが普段から使っているものである限り，小手先の法則に意味はないのだ。

普段，乏しい語彙で，その乏しさを表情や身振り手振り，互いのあいだでのみ通用する隠語や造語や流行語でカバーしている人たちがひとたび文章でなにかを表現しようと思うと，途端に，言葉だけで自分の意見を表現することがいかに難しいかを知るだろう。

我々は普段，言葉によるコミュニケーションをしていると勘違いしがちだが，実のところ，言葉以外のちょっとした表情やら言い回しやら，声のトーンや，つっかえ方により多くの意味を求めているのだ。

確かに，学校の先生の気に入る感想文の書き方や，論理的な（に見える）レポートの書き方や，上司に受ける報告書のコツなんてものは存在する。人を感動させる文法も存在する。しかし，そんな対症療法にたいした意味はない。どんなに型を知ったところで語彙の不足はどうにもならないし，んなものはすぐばれる。人を感動させるテクニックを駆使したところで，“一杯のかけそば”のように下品な話がまた増えるだけだ。

それならば，自分たちの新しい日本語を作っていけばいいではないか，と。つまり，精神論を踏まえ，技術は生まれてから10数年，20数年のあいだに培ってきたものを流用しようと，そういうわけなのである。

ここまで書いて読み返してみると，自分の文章に悪文が多いことに気づく。だから，文章を書くときの最大のポイントを掲げて終わりにしよう。

“自分のことは棚に上げろ”

自分の考えを持つとは

どんなに文章がうまくても，内容がなくては週刊誌の埋め草にしかならない。

内容というのは，書いた人の視点や考え方がどれだけ読者にとって面白いかである。新鮮であったり，捻ってあったり，予想外の展開をしたり，といったことだ。

いまの時代，ひとつのことだけを極めようったって（ひとつの手ではあるが），そうはいかない。上には上がいる。じゃあ，どうするかというと，自分の得意な分野の視点でほかのものを見てみる，とか，自分の得意な分野とほかの分野を合体させて新しいものが見えてこないか探ってみるのがいちばんてっとりばやい。

人は自分の知らないことは書けない，の法則によって，そうそう独創的な考えなんて出てきはしない。それでも人の経験，人の知識はそれぞれがユニーク（唯一）なものであるからには，自分の考えを持つことは可能である。あらゆる事象を自分で解釈しようと思っていれば，おのずから自分の考えというものがどういうものかはわかってくる。隠れた意志を知れ，ということだ。

特集1

日本語を処理するための序章

X68000の日本語環境を見る

我慢せずに使うWP.X

Nakano Shuichi
中野 修一

X68000に標準で付属するWP. X。マウスオペレーションを重視しすぎたためか，バランスを崩してしまったように見えるところもある。ここでは日本語フロントプロセッサASK68Kとともに賢い使い方を摸索してみよう。

タコと呼ばれて久しいX68000のワープロですが，確かにちょっと変なところがあります。それは日本語変換フロントプロセッサASK68Kに起因するもの，ワープロ自体の仕様に起因するものなどさまざまです。

まず，ASK68Kを見てみましょう。わからないものまで無理矢理変換しようとするので，変換後に手動操作が多く必要になるとか，2文節最長一致法というアルゴリズムで連文節変換を行うにも関わらず，辞書の作り方が熟語変換的なものでしかないといった点があります。

私たちが使う日本語には漢字の部分とカタカナ，ひらがなの部分があります。当然，ひらがなで書くべき単語というものもあるわけですが，600Kバイト以上にもおよぶ膨大な辞書の中身はほとんどが漢字を使った単語なのです。基本的に漢字にする必要がある部分でのみ使用し，変換の必要がない部分は手作業という方針のようです。

漢字変換システムとすればこれでもいいような気もしますが，日本語処理システムとすれば問題があります。なにしろ，ひらがなやカタカナで表したい単語がすべて変な当て字で出力されるのですから。これは，ひらがなの単語は端から登録していくことである程度解決されます。もっとも利口なのは必要な部分しか変換しないということでしょう。

また，文法解析が甘く，当然推測できる単語が変換できなかったりもします。いくら文節を切り直しても必要な単語が出てこない場合もあります。たとえば，単に，

美しさ

とすると，

鵜ツクシさ

と変換されてしまいます。形容詞の語幹に「さ」をつけると名詞扱いされるということが徹底されていないようで，文章中でもたまにおかしくなります。形容詞の語幹に「げ」をつけた形容動詞形の場合はほぼ全滅です。わざわざ「明るさ」のように名詞で別登録されている場合もあります。

そのほか，動詞の連用形は名詞として使えるというのも利用されていないようです。結果として辞書が肥大化するわりには取りこぼしが多くなっています。

一方，標準ワープロでは，マウスの使用を強制されることやコントロールキーがほとんどサポートされていないこと，禁則の際のカーソル移動がおかしい，変換キーを押し続けても次候補が途中で止まってしまうなどの問題点があります。細かいことを挙げればきりがありませんので適当なところでやめておきましょう。

では，X68000の日本語環境はどうしようもないのでしょうか？

幾多の問題点にもかかわらず，Oh!Xの誌面のほとんどはX68000の標準ワープロで作られているというのも事実です。なぜで

編集者御用達「無変換学習なし」

遅いといわれたASK68Kもver.2.0になってからはかなり高速になった。特にフロッピーディスクやハードディスク上での使用速度は格段に向上している。X68000の標準ワープロではスクロールの高速さとあいまって速度だけは快適な環境が得られている。が，ワープロはそのままではちょっと使いにくい。

たとえば入力時に枠が開くと位置揃えや修正の際に邪魔になる。不便だ。

ASK68Kの辞書はなにがなんでも漢字にしたがる傾向がある。しかしながら，普通の文章には漢字も多いが，ひらがなだって多いものだ。そこで無変換モード。漢字が必要な部分でだけ漢字変換を行い，漢字が必要ないときにはひらがなのまま確定してしまうわけだ。これなら最小限度の変換作業ですむ。

お，なんとなく普通のワープロっぽくなったぞ。おお，しかも，ひらがな部分の確定には，いにしえの即戦力と同じTABキーが使える（タブを設定してないときは)。

●学習なんかいらない

無変換モードは辞書の先読みをしない。その分，入力が軽くなり，変換が重くなる。しかし，漢字の現れる部分でしか変換しないのだから，変換操作のいらないひらがなの分だけ変換は軽くなっているとも考えられるわけだ。これで速くなった……，いや，変換終了時に余計に時間がかかるぞ。

さて，思い切ってここで辞書のプルダウンメニューを開き，辞書学習なしに設定する。途端に軽くなったはずだ。学習なしで大丈夫なのかという心配もあるだろう。実はこの学習あり/なしというのはASK68Kのディスク学習，メモリ学習に対応している。

さて，ここでワープロを立ち上げ，無変換学習なしのモードにする。そして，適当に文章を変換してみる。変換モードを逐次などに変えてみる。なにも起きない。辞書を学習ありに設定する。なにも起きない。変換モードを無変換に戻す。するとメモリ上の学習内容がディスクへ転送されるのがわかる。ゆえに，文書変換時に学習モードにする必要はない。

もっとも，ディスクに学習しなくとも，しばらく学習つきで鍛えたあとなら学習なしでも十分に使える辞書になっているはずだが。

●無変換の注意

これでかなり高速なワープロとして使うことができるようになったわけだが，無変換には意外な罠がある。メモリ学習が重くなってくると「変換ウィンドウを開いたまま，次の文字の入力をもって確定に変える」場合にローマ字かな変換を失敗することがあるのだ。たとえば，「ここで」と入力したいのに「Kおこで」のようになることがある。いったんこうなると，同じ症状が多発するようになる。

これはASK68Kをセットアップする際に，DEF CONT＝1に設定することである程度抑えられるようだ。それでも症状が出たときは変換モードを一度ほかのモードに切り替えるとよい。当然のことながら変換時に確定していく癖をつけるのがいちばん確実な対処方法だ。なお，普通のキー設定なら，リターンキー，XF2，XF5，CTRL-Mのいずれかで確定される。

＊　＊　＊

さて，たくさんのマシンが使え，隣近所の編集部に声をかければ，たいていのワープロソフトは試用できる，そういった環境でも日々WP.Xは使われているわけだ。それは，いまやMS-DOSよりHuman68kのほうが快適だということ，そして，ひたすら入力していく作業はともかく，すでにある文書を編集していく場合にはWP.Xは非常に扱いやすいということによる。

あとは新感覚のワンバウンドスクロールバーほか，数十個の不都合さえなければいいワープロになるのにねぇ。 (U)

しょう？　それは，つまりここで挙げた問題点はちょっとした工夫や辞書の鍛え方でかなり改善されるということです。

せっかくの日本語環境，少しでも上手に使うことを考えてみましょう。

ASK68Kのカスタマイズ

最初にすべきことはASK68Kのバージョン確認です。必ず2.01を使用してください。初期バージョンは時折，人生をはかなんで遠い旅に出るという典雅な趣味を持っていましたが，最近のものはそう世間知らずでもありません。

さて，ASK68Kのバージョン2.00以降では環境ファイルを設定することにより漢字変換時のキー操作をユーザーがカスタマイズすることができます。表1がもっとも標準的なENV1.ASKです。これ以外にもいくつかの定義例が見られます。

まず，このASK68Kの標準設定を見てみましょう。X68000のキーボード自体が日本語入力を意識して設計されたものであり，ASK68Kもこれにあわせて作られたためか，相性はかなりよいようです。特に文節の移動や切り直し（要するにXF1,2の使い方）はずば抜けて優れているように思えます。このあたりはわざわざキーボードに馴染まない方式に変える必要はないでしょう。

もっとも重要なのは，最終行のDEFCONTを1にすること，です。これはかな漢字変換時の1行ウィンドウが未確定でも次の文字列を入力できる機能です。これを設定せずにHyperwordを使うとちょっとうっとうしいことになります。標準ワープロでもこれをしておかないと無変換モードでの変換中にローマ字変換を失敗することが多いので必ず設定するようにしてください。そのほかエディタなどを使う場合もこの変更が有効です。

あと，XF5キーというのは人間に押せるキーではありませんから，ENTER（確定）の機能は再定義するか，リターンキーで代用します。そのほかは好みでDEFECHOを設定するとか，その程度しかいじる必要はないと思います。

このファイルをいじれば，ファンクションキー，コントロールキー，変換キー，カーソル関係キーなど，ほぼ思いどおりの変換キー設定ができます。ただ，TABキーについては機能割り当てができないようです（バグ？）。

さあ，これで最低限の設定は終わりです。

辞書はどこにありますか？

ハードディスクをお持ちですか？　辞書はどこにありますか？

Human68k ver.2.0とASK68Kver.2.0以降は変換速度が大幅に向上していますので，RAMディスクは必ずしも必要ではありません。辞書の先読みをしない無変換モードでさえ，ハードディスクならRAMと比べて遜色ない速度で変換を行ってくれます。

ディスクドライブの構造とディスク管理の都合から，高速アクセスしたいものは連続領域に置くのが基本。ハードディスクに辞書を置く場合は，専用の辞書ドライブを設定することをおすすめします。フォーマットの際に2Mバイトもパーテーションを切っておけば十分でしょう。

ASK68Kのメイン辞書は内部にフリーエリアを確保していますから再編成するまで大きさは変わりません。しかし，サブ辞書はどんどん大きくなりますから，途中にほかのファイルが割り込まないように隔離しておくわけです。確実に連続領域に配置するには，真新しいディスクに辞書ファイルだけを置いておくようにするのがもっとも効果的なのです。

フロッピーベースで使用している場合なら，BドライブまたはRAMディスクに辞書を置くことになります。当然，フロッピーに比べ，RAMディスクは非常に高速です。ASK68Kver.2.0以降ならフロッピーでも十分使えます。

ただ，もはや2Mバイトで辞書をRAMディスクに置くのはおすすめできません（RAMがもったいない）。標準ワープロ専用で起動する場合ならいざ知らず，RAMディスクに割り当てられるメモリがあるなら，作業用ドライブとして使うほうがなにかと便利ではないかと思われます（使用法にもよりますが）。

辞書の鍛え方

変換されなかった単語は片っ端から登録する，これが基本です。同様に勝手に変換されたくない単語も登録します。変換直後は学習効果のため登録なしでもうまく変換してくれますが，それ以後も変換してくれる保証はありません。

WP.Xの単語登録は範囲指定して登録キーを押すだけですので非常に簡単です（キーボードだけで操作できる数少ない機能のひとつ）。WP.X上では名詞関係の登録しかできませんので，動詞や副詞などはとりあえずメモしておき，あとでまとめて登録しましょう。

ASK68Kの辞書ディスクに収録されている単語は表2のとおりです。数字関係の部分をダンプしてみれば数詞の0個という

表1

ENV1.ASKの設定例	機能説明（0/1に対応）
BEGIN=CTRL+XF1	フロントプロセッサ起動キー
END=CTRL+XF1	変換モード終了キー
XFER=XF3	変換キー
ENTER=XF5	確定キー
TYPE=F10	一括変換/逐次変換切り替えキー
DEL=DEL	1文字削除キー
RIGHT=RIGHT	カーソル右移動キー
LEFT=LEFT	カーソル左移動キー
HOME=HOME	カーソルを変換部の先頭へ
CLR=CLR	変換文字列消去キー
CODE=F7	コード体系変更キー
LEARN=F9	学習機能切り替えキー
DIC=F8	辞書の変更キー
HIRAKATA=XF4	ひらがな/カタカナ変換キー
ZENHAN=SHIFT+XF4	全角/半角切り替えキー
NEXTKOUHO1=RIGHT	次候補表示キー
NEXTKOUHO2=XF3	次候補表示キー
NEXTKOUHO3=SP	次候補表示キー
BACKKOUHO1=LEFT	前候補表示キー
BACKKOUHO2=SHIFT+XF3	前候補表示キー
BACKKOUHO3=NULL	前候補表示キー
NEXTBLOCK=DOWN	次の候補群呼び出しキー
BACKBLOCK=UP	前の候補群呼び出しキー
SHORTER=SHIFT+XF1	文節縮小キー
LONGER=SHIFT+XF2	文節拡大キー
NEXTBUN=XF2	次の文節へ
BACKBUN=XF1	前の文節へ
ECHO=F6	エコーモード切り替えキー
DEFECHO=0	カーソル位置での変換しない/する
DEFROME=1	ローマ字かな変換しない/する
DEFZEN=1	全角ロックで起動しない/する
DEFHIRA=1	ひらがなロックで起動しない/する
DEFINS=0	インサートモードで起動しない/する
DEFALL=1	逐次/一括変換モードで起動
DEFMEM=1	ディスク学習/メモリ学習で起動
DEFCONT=0	確定必要/不要モードで起動

のは実に納得できます。

一説によると，名詞を形容動詞複合名詞にするとヒット率が上がるといいます。試しに固有名詞を除くすべての名詞を形容動詞複合名詞に変更してみました。若干，助詞の判定がよくなったかもしれませんが，前の辞書がすでに助詞対策されていたので（「と」，「の」，「が」などの助詞を名詞として登録しておく），あまり目立った違いは見られませんでした。とりあえず新しく登録する単語だけ形容動詞複合名詞にしておけばいいでしょう。

こうして，漢字変換も使い込んでいくうちにかなりヒット率が上がっていきます。なお，新しく登録された単語は辞書内部のフリーエリアに格納されます。フリーエリアがなくなると，変換時に異様に時間がかかってきます。辞書はまめに再編成しましょう。

あとは無駄に思える単語は削ります。あなたは辞書の中身を見たことはありますか？　DICM.Xを起動し，単語一覧を選択してください。たとえば，開始位置を“あさひ”，終了位置ではリターンを入力してみます。なにか疑問を感じませんか？　次に開始位置を“さんきろめーとる”，終了位置では単にリターンを押してください。……。

さあ，削れるものはみな削りましょう。村田氏のユーティリティが便利です。

もっと速く

スクロール速度を上げる手っ取り早い方法として，キー入力のウエイトを調整する手があります。方法は簡単，SWITCH.Xを起動してメモリスイッチのFirstKeyとNextKeyを変更します。Oh!XではFirst Keyを1，NextKeyを0くらいにするのが流行っています（キーウエイトを変更すると，ちょっとだけYet Another Columnが有利になるかもしれません）。

ただし，この変更を行うと，キーを長時

表2　辞書ディスクver.2.0の内容

品詞	語数
カ行五段活用動詞	236
ガ行五段活用動詞	54
サ行五段活用動詞	509
タ行五段活用動詞	59
ナ行五段活用動詞	1
バ行五段活用動詞	32
マ行五段活用動詞	240
ラ行五段活用動詞	668
ワ行五段活用動詞	238
サ行変格活用動詞	155
カ行変格活用動詞	2
上・下一段活用動詞	1033
形容詞	456
形容動詞	44
形容動詞複合名詞	1812
サ変複合名詞	6484
名詞	25840
単漢字	3351
人名（姓）	5967
人名（名）	3371
地名	4587
団体名	1324
物の名称	26
数詞	0
数字	0
接尾語	0
感動詞	24
接続詞	37
副詞	602
連体詞	53

エディタ≧日本語ワープロ あるいはMicroEmacs賛歌

Izumi Daisuke
泉　大介

世の中にはアンチワープロ派といわれる人々が存在する。彼らは日本語の文書を書くのに決してワープロを使わない（状況に応じて否応なくワープロを強制されることはあっても，束縛から解放されるやいなや彼らの魂はその故郷へと帰っていく）。作家が使い古した万年筆を執筆の友とするように，手に馴染んだエディタの元へと回帰していくのである。

エディタはプログラムを入力・編集する道具として進化してきた。作成されるFORTRANのプログラムはときには数万行に及び，LISPのプログラムはゾッとするほどの括弧とインデントを従えている。これら強者と対等に渡りあうだけの能力を人々は求め，エディタはそれに応えて見事強敵を打破してきた。

そのなかで「これはエディタというよりむしろ環境と呼んでいい」と高く評価されているものがある。UNIXの世界であまねく知られたEmacsである。高機能さ柔軟さに加え，特異なキー割り付けが，いったん染まってしまった者をほかのエディタから遠ざけ続ける。

これほどまでに人々を魅了する秘密のひとつは，Emacsが備えるすべての機能がユーザーに開放されていることだろう。キー割り付けの変更はもとより，自分に必要な機能をプログラムしそれを好みのキーに割り振るなどが簡単にできる。マクロではない。Emacs Lispである。

Emacsが環境だという賛辞は，専用のエディタを構築できるという魅力だけにとどまらない。編集中のテキストはバッファと呼ばれる領域に読み込まれディスプレイに表示されているが，なんと，このバッファの中でシェルが動くのである。キーボードからの入力は普通のテキストと同じようにバッファに入力され，リターンキーを押すと同時にコマンドとして実行。その実行結果も同様にバッファ内に残る。必要ならEmacsの高度な編集機能を駆使してそれらを加工し，もう一度シェルへのコマンドとして実行できるのである。1行入力しかできなかったプログラミング言語が，たちまちフルスクリーンエディタつきの言語へと変身する。

●MicroEmacs

この強力なエディタEmacsは，そのサブセットがパーソナルコンピュータにも移植されている。電脳倶楽部でも配布されたMicroEmacsである。シングルタスクのHuman上で動くため，バッファ内でシェルを起動することはできないが，ユーザーにほとんどの機能を開放するという姿勢は堅持されている。Lispではないがマクロ言語をサポートしており，足りない機能をプログラムによって補うことが可能なのである。

たとえばGNU Emacsにはdiredというコマンドがある。これはディレクトリエディタを略したもので，任意のディレクトリを表示しながら，編集するファイルをカーソルで選択するコマンドである。サブディレクトリのアップダウンはもちろん，ファイルの削除までサポートしている。標準のMicroEmacsにこの機能はないが，私のMicroEmacsではdiredが動いている。そう。マクロを使って定義したのだ。その他，Humanのコマンドをバッファ内から直接起動するなんてマクロもある。こうしてすでにMicroEmacsは私にとってなくてはならない環境に成長した。

手塩にかけて成長させた高機能なエディタは体の一部といってもいい。必要に応じて作成したマクロは，自分の使いやすいキーに割り振ってある。既存のワープロを使うことは，これらすべてを破棄することなのである。

●MicroEmacsの日本語処理

このように強力なMicroEmacsだが，難点もないではない。その最たるものは日本語入力時に字詰めを決められないという点であった。バージョン3.9には指定された文字数よりはみ出す単語は入力と同時に次の行へ送られるWRAPモード（英文の自動字詰め機能）が備わっていたが，これは日本語では動作しない代物だった。

バージョン3.10（「さんてんじゅう」と読む）で改善が加えられ，X68000用のMicroEmacsではこのWRAPモードが日本語でも使えるようになったのである。単に文字数を揃えるだけでなく禁則処理まで行ってくれる。編集を指向したプログラムが字詰めという入力指向の機能を手に入れたのである。ただ残念なことに桁揃えされた各行の最後には改行が入ってしまうので，テキストファイルにする場合にはこれを取り除かなければならない。

なぜワープロで文書を作らないのかと疑問をお持ちになる方もいるだろう。私にとって，ものを書くときに大切なのは思考をとぎれさせないことである。入力した文字は読み直され，吟味され，修正され，そして文章となる。推敲のたびにマウスに手を伸ばさなければならなかったり，カーソルキーまで手を伸ばさなければならなかったり，ファンクションキーからメニューを選ばなければならなかったりするのは推敲のリズムを乱す以外のなにものでもない。万年筆で小説を書くときに，削除する部分を修正液で消すだろうか。カバーする大きさの原稿用紙を切り抜いて貼り付けるだろうか。

もしあなたが華麗な装飾を施した，見る人を魅惑するようなものを作りたいと思うのならワープロを使うのがいい。もしあなたが罫線を駆使した体裁のよい表を作りたいと思うのならワープロを使うのがいい。しかし，もしあなたが本当に文章を「書きたい」と思うのなら，MicroEmacsを使うべきである。

図1　X68K_M.DICの構成

000000H～001FFFH　インデックス領域
（1E00以降にフラグ領域？）

002000H～　辞書の本体

ただし，1ブロック＝1024バイト

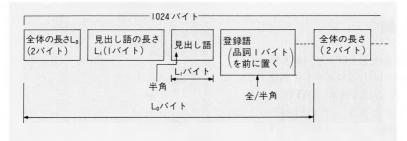
図2　X68K_M.DICの辞書ブロック内部構造

全体の長さL_0は下位・上位の順で記録されている。また1024バイトに満たない部分は“00_H”で埋められる

図3　見出し語コード

上位＼下位	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1																
2	␣	!	“	#	$	%	&	‘	(	)	*	+	,	－	.	／
3	0	1	2	3	4	5	6	7	8	9	:	;	<	＝	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	¥	]	^	_
6		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	‾	
8																
9																
A		ァ	ア	ィ	イ	ゥ	ウ	ェ	エ	ォ	オ	カ	ガ	キ	ギ	ク
B	グ	ケ	ゲ	コ	ゴ	サ	ザ	シ	ジ	ス	ズ	セ	ゼ	ソ	ゾ	タ
C	ダ	チ	ヂ	ッ	ツ	ヅ	テ	デ	ト	ド	ナ	ニ	ヌ	ネ	ノ	ハ
D	バ	パ	ヒ	ビ	ピ	フ	ブ	プ	ヘ	ベ	ペ	ホ	ボ	ポ	マ	ミ
E	ム	メ	モ	ャ	ヤ	ュ	ユ	ョ	ヨ	ラ	リ	ル	レ	ロ	ヮ	ワ
F	ヰ	ヱ	ヲ	ン	ヴ	ヵ	ヶ									

間押し続けた場合に画面のスクロールがキーに追いつかずキーを離してもしばらくスクロールが続くようになってしまいます（要するにバッファがたまる）。どうしようもないときはシフト＋ブレイクで止めてください。

ASK68Kの辞書構造

1988年2月号でX68000にX1turbo用の辞書，WORD POWERを移植するという記事が掲載されました。同時に投稿者の長井氏が解析したX68000の辞書構造が発表されています。

辞書の構造について，ざっと解説しておきましょう。辞書はインデックス部分と本体に分かれ，辞書本体には登録語が入っています（当たり前）。これは読みがな順に並べられており，頭から1024バイトごとにブロック分けされています。

辞書の先頭にはインデックスがあり，ダンプしてみると各ブロックの最初の見出し語（の8文字分）が整然と並んでいるのがわかるでしょう。ただし，これはASK68Kの内部コードで記述されていますので，読むためにはコードを変換しなければなりません（図3）。

辞書の先頭の2000_Hバイト目から辞書本体が始まります。この部分は，ある読みに対する熟語のグループが集まったものです。

最初の2バイトがその読みのグループ全体の大きさを表し，次に見出し語の大きさ（1バイト），内部コード化された見出し語本体が続きます。そして登録語です。1バイトの品詞情報の後ろにシフトJIS（＋ASCII）コードの登録語が並んでいます。同じ読みに対して複数の語が対応するときは，後ろに品詞コードと単語が続いていきます。品詞コードは$1E_H$以下の通常コントロールコードとして使われる部分に配置されていますので，登録語の分割は簡単です。

単語の読みからインデックスを検索すれば，辞書のどのブロックに目的の語が入っているかがわかります。そのブロックを読み込み，対応するグループを探して候補群としています。どれか単語が選択されるとその単語をサブ辞書に登録して「学習」をします。

プログラマーズマニュアルでは日本語FPのファンクションコールが公開されています。ユーザーにも日本語処理プログラムを作れる可能性は十分あるわけです。

＊　＊　＊

罫線関係，記号入力や外字作成などは完璧といっていいでしょう。字詰めの変更がこれほど簡単なワープロはほかにまずないでしょう。加えてマウスに割り当てられた機能の使いやすさは国産ソフト中でも屈指のものがあります（その分，キーボードの使い方が下手なのが残念です）。

ASK68Kは使い込んでいくことでどんどん確実な変換をするようになります。鍛えられた辞書はユーザーの財産となります。皆さんも，自分なりにASK68KとWP.Xをうまく使いこなしてください。

図4　登録語/品詞コード

品詞	コード	品詞	コード
動詞（カ行5段）	01	サ変複合名詞	10
〃（ガ　〃　）	02	名詞	11
〃（サ　〃　）	03	単漢字	12
〃（タ　〃　）	04	人名（姓）	13
〃（ナ　〃　）	05	〃（名）	14
〃（バ　〃　）	06	地名	15
〃（マ　〃　）	07	団体名	16
〃（ラ　〃　）	08	物の名称	17
〃（ワ　〃　）	09	数詞	18
〃（サ行変格）	0A	数字	19
〃（カ行変格）	0B	接尾語	1A
〃（上下一段）	0C	感動詞	1B
形容詞	0D	接続詞	1C
形容動詞	0E	副詞	1D
形容動詞複合名詞	0F	連体詞	1E

表3

読み	登録語	品詞
アサヒ：	朝日	／名詞
	朝日	／人名（姓）
	朝日	／団体名
	旭	／名詞
	旭	／地名
	旭	／団体名
	アサヒ	／団体名
サンキロメートル：	三KM	／名詞
	三km	／名詞
	三キロメートル	／名詞
	3KM	／名詞
	3km	／名詞
	3キロメートル	／名詞

それでも私はHyperwordを使う

Ogikubo Kei 荻窪 圭

私，荻窪圭は，Hyperwordで原稿を書く。人はそれを見て驚く。つまり，"遅いソフトはそれだけで存在価値がない"といった神話が生きているからである。

Hyperwordは確かに遅い。スクロールや，変換時の反応が遅い。しかし，「狭い紙面，そんなに急いでなにを打つ」である。おっと，反論がきた。「迫る締め切り。そんなにのんびりしてられない」。困ったなあ。でも，そうまでして速いものがほしいですか？

Hyperwordのカスタマイズ

Hyperwordは非常にわがままなワープロ。使用する環境が悪いととても使うに耐えないし，環境を整えとけば，それなりの力を発揮する。機能の貧弱さを速度で補うのも手だが，その逆もまた有効なのだ。

Hyperwordを使うには2つのポイントがある。ひとつはどれだけHyperwordのために資源を割けるか，であり，もうひとつが，カスタマイズである。

前者だが，Hyperwordは実に頻繁に中間ワークファイルを作成する。こればかりはいかんともしがたい。よって，RAMディスクを使用する。私は256KバイトほどそのためにRAMを使っている。RAMディスクがIドライブだとすると，立ち上げ時オプションに"-wI"をつければいい。さらにHyperwordはプログラムサイズが迷惑なほど大きいので，ハードディスクに入れると起動時のイライラが少なくてすむ。

続いて後者だが，Hyperwordは環境ファイルを持っている。私の使っているものが図1である。こいつがポイントだ。

右半分にコメントのついているヤツがシステムについてくるキーバインドである。ただ数字を書いてあるのが私の追加した分だ。数字は機能コードであり，Hyperword独自のものである。96が行末，95が行頭へのカーソル移動，106はカーソル位置から行末まで削除，18と19は1行ロールアップ/ダウンである。Hyperwordがあらかじめ用意している機能だけだが，こうしてカスタマイズ可能だ。

ユーザーカスタマイズしたキーはXF1キーファンクションとなり，あらかじめHyperwordがショートカットキーとして用意したものはCTRLファンクションとなっている。エディタを使い慣れている我々にとってこれは非常に不便である。

ここで登場するのが"-x"オプションだ。このオプションをつけて起動すると，ユーザーカスタマイズした機能がCTRLファンクションになり，HyperwordのショートカットキーがXF1になる。便利である。

最低，このくらいの環境が必要だ。標準ワープロがCTRLファンクションをほとんどサポートしていないことを考えれば，これだけでもうれしい。

私はこう使う

さて，この環境でHyperwordを立ち上げよう。すると，横19文字で(半角だと38字)"新規文書1"ウィンドウが開く。Oh!Xの原稿は横19字が基本なので，標準書式として登録してあるのだ。

新規文書1にタイトルを書き，おもむろに，「本文」と「メモ」の2つの目次を作る。本文の下に思いついたものを片端から書きなぐり，XF1＋Oで本文シートを開いて，メモをもとに本文を書く。

まだHyperwordに懐疑的な向きは多い。確かに，印刷系が弱いとか（縦書き印刷ができない），ファイル操作系が弱いとか(ディレクトリ名までは自分で打たなければならない）いう欠点はあるが，文章作成支援道具と思えば，それほど気にはならない。

Hyperwordはマウスを使ったオペレーションがウリだと思われているが，そうではない。マウスは気が向いたときやウィンドウ操作にしか使わないのが常だ。

たとえば範囲指定はXF1＋カーソルキーを使う。削除したいときはDELキー，クリップボードへカットしたいときはXF1＋X，コピーはXF1＋C，ペーストはXF1＋Vだ。これはMacintoshと一緒。ロードはXF1＋L，終了はXF1＋Qだ。範囲指定して別の文字を入力すると（多少もたつくけど），新しい文字に置き換わるのだ。覚えていない機能やキーバインドされていない機能はファンクションキーでプルダウンメニューが開くのだ。ファンクションキーが遠くていやなら，どっかのキーにバインドしておけばいい。

文書の管理も，アイデアプロセッサの機能を使えば，複数のものをひとまとめに管理できるので，たとえばパソコン通信でもらったメールの管理なんかも簡単。

マルチウィンドウも便利。いろんな文章を横目で見ながら書けるから，上下2分割程度とは雲泥の差だ。

しかも，どこで悪いことをしているのか，全角キーをOFFにすれば，自動的に英数字入力モードになるので，遠くのローマ字キーまで指を伸ばさなくて済む。

というわけで，私はHyperwordを使っている。変換時のキー反応を速くしろ！　とか，削除/挿入が遅すぎるぞ！　という文句はとてもあるのだが，とりあえず，付属ワープロよりは使いでがあるのだ。私としては印字機能を別プログラムにして，縦書きとか2段組みをサポートしたちゃんとしたやつにして（本体にはテスト印字程度の簡単なものしか与えない），プログラムを小さくしてもらいたい。

速度も，もっと速くできる気がする。さらに，テキストファイルの読み書きをもっと速くして，カーソルの現在位置がわかるようにして，ついでにヘルプ機能もつけてほしい。シートごとに書式も変えたい。書式設定画面も使いにくい。

しかし，私はHyperwordを使う。PC-9801で書くときは（たとえばシムシティーの原稿を書くときはX68000が街作りで忙しかったのでPC-9801を使った），VZエディタを使う。だから速い環境も知っているのだ。それでも，速さがすべてではない。私の目にHyperwordのマルチウィンドウは優しい。

図1

```
*----------------------------------------------------------------------*
* HPW.KEY --- Definition file for power key. This file define some of  *
* WordStar compatible operations. See "3.9.2 Power key operation" of   *
* HyperWord manual for the definition of power key, and see "appendix 4 *
* List of operation" for the code of each operation.                   *
*----------------------------------------------------------------------*
{EditKey                                 * definition for editting key
   e = 1                                 * [XF1]+e: move cursor up
   x = 2                                 * [XF1]+x: move cursor down
   s = 3                                 * [XF1]+s: move cursor left
   d = 4                                 * [XF1]+d: move cursor right
   r = 14                                * [XF1]+r: roll up
   c = 15                                * [XF1]+c: roll down
   h = 24                                * [XF1]+h: back space (BS)
   g = 25                                * [XF1]+g: delete character (DEL)
   v = 26                                * [XF1]+v: change store mode
                                         *          of character (INS)
   i = 27                                * [XF1]+i: tabulation (TAB)
   m = 29                                * [XF1]+m: carriage return (CR)
   p = 96
   q = 95
   k = 106
   w = 18
   z = 19
   }
* XF1キーとなっているが、-xオプション付起動をするため、
* CTRLと読み替えてください。                        荻窪
```

特集1

日本語を処理するための序章

雷語1号はどうなるのか？

ホメオスタシスへの道

X1turboZで電脳倶楽部の原稿を書く祝一平氏も，なんとかならないものかと思い始めた今日この頃。求むべきは，サクサクと，ただひたすら頭の中に浮かんだ文章を入力できるバランス状態である。かくして雷語1号はながーい発進準備に入った。

Iwai Ippei
祝 一平

コンピュータのお仕事が「計算」から「情報処理」になり，そのなかで日本語ワープロが重要な機能となって，はや四半世紀である（かどうかはよく知らないが，まあそんなもんだろう）。そーゆーわけであるから，X68000が発売開始になったときに，それまでX1turboで即戦力を愛用していた私は，X68000用に使えそうなワープロが出たならば，たちまち乗り換えるつもりでいたのである。

んが，ああそれなのに幾星霜。ふと気づけばなんということであろう，もう3年以上もたつとゆーのに，あいもかわらず文章を書くときには，よっこらせとX1turbo Zで即戦力を立ち上げている私でわぬわいくわっ！

まあ，誤解のないように言っておくが，X68000用のワープロが全部ダメというわけではなく，私にとって即戦力よりも使い易いと実感できるものがないということである。こないだまではHyperwordに若干の期待があったわけであるが，残念ながら速度の点で見送りである。速ければよいというわけではないが，思考のリズムが狂わない程度に速くなくては困るのだ。

てなとこで，思うに，ほかになんか出るとしても，あとで述べる諸々の事情により，私の要求を満たすものが出てきそうな気配は今のところはないようなのである。うーん，困った。

そこで，あの頃に帰ってみようかと思う。「ないのなら，自分で作ればいいのさ」と，なんの疑問も持たずに信じていたあの頃にである。

日本にパソコンが出現して10年ぐらいたったようだけど，その間には普及台数の飛躍的な伸びもあったし，それにともなってソフトウェアの技術レベルもどんどん上がっていって，そしていつしか「自分で作る」ということが，ものすごく，マニアックなことになってしまった。

でも，今だって「ないのなら作ってしまえばいいのさ」という事実に，変わりはないはずだろう（そりゃそーだ）。

で，考えようによっちゃ，X68000のソフトを作るための環境はものすごく整っている。DOSは標準で付いてくるうえに，バージョンアップなんかで別売のものを購入するにしても10,000円を切る値段で出てくるし（これって冷静に考えてみりゃ，トンでもないことなんだぜ），辞書は標準で付いているからそれを使うようにすればいいし（そういえば，昔，辞書をほかのソフトからパクッて大騒ぎになったワープロがあったっけ），XCのver.2.0もそろそろリリースされるであろうし。

そしてホメオスタシス

実をいうと，私は密かに「帝国主義の最終形態はソ連型共産主義であり，独占資本の最終形態はソ連型共産党である」という理論を持っていたのであった。もしもこの理論を一昨年ぐらいに発表しておけば，今頃はおニャン子政治学者の舛添先生といっしょにマスコミでウハウハしてたかもしんない。おしいことをしたなぁ（後知恵度：60%）。

まあ，とにかく，やっぱり独裁とか独占とかゆーことは，あんまりよくないことなのである。そーゆー意味からも，そろそろX68000になかなかのワープロが出てきてほしい頃ではないか。やっぱ，ものごとつーもんは，多様性にこそ真の安定と発展があるのだのだ。

そこで，「謎の日本語ワープロ」の暫定仕様を発表するわけであるが，ただし，あくまでさわりだけである。なぜかというと，もしも全貌を公表してしまったのならば，ヅャヌト某とかいう会社の社長が反復横跳びをしてしまうほどすごいからである。まあ，それぐらい画期的なワープロになる予定なのである（ハッタリ度：測定不能）。

まずはソフト名であるが，前々からホラっておいたように「雷語（サンダーワード）1号」である。どどどどど。

このソフトの基本は，あくまでも「質実剛健」にある。軽快さを失わず，かといって必要な機能は決しておろそかにしないのである。多機能ではなく高機能なのである。プロテクトはかかってないし，増設メモリなしでも動く予定だぞ。どうだまいったか。うりうり。

さっそく起動方法であるが，当然ながら，最初にHumanが立ち上がるな。んで，AUTOEXEC.BATが環境変数なんかを設定したあとで，自動的にワープロに入ってもいいのだが，べつにそーゆーふうな起動には限ってないから，なんかほかのことをやったあとで，あたかもED.Xを起動するように，

TW1 ファイル名［CR］

でよいのである。

さて，肝心の諸機能であるが，まずこの際ハッキリしておきたいのが，

マルチウィンドウ機能なんかねーぞ

ということである。

実に嘆かわしいことであるが，最近は発売日の何カ月も前からドカドカと広告を打つということが常習となっている。その結果として，多くのソフトはカタログスペックの充実ばかりに走っているのではないだろうか？　その結果マルチウィンドウとかの飛び道具に走る傾向が絶えない。困ったものである。ソフトウェアの真実はハッタリにあらず，便利にあり。質実剛健こそ永遠の王道なのである。そこで改めて問い直したい。

エディタでオーバーラップするウィンドウが沢山あったとして，いったい何が面白いんだ？

アイデアプロセッサなんていったって，結局は書き手の脳ミソのレベル以上の文章ができるわけがないじゃないか。

そりゃあ，でかいハードディスクを積んだバキバキの32ビットのワークステーショ

ンで，OSに仮想記憶とかがあるならば，それぐらいのものがあってもいいけどさ。

とにかく，オーバーラップ処理はかなり重い処理なのだ。そこで，質実剛健な雷語1号としては，変に重いミエ機能は排除し，とりあえず水平画面分割だけをサポートするのである。これはまあ，簡単に表現すれば，画面を上下に2分割し，2つのファイル(もしくは1つのファイルの別々の部分)を同時に表示し，編集するというものにすぎない。ただし2分割だけではちょっと不便なので，そうだな，最高で8分割ぐらいまではできるようにしておこう。1ファイル4行だな(行数の増減可)。本当は32分割や，1/4角文字を使っての64分割ぐらいまでやってみてもよいのだが，やっぱり何かと不便だろうからこれでいいのだ。うむうむ。

それからであるが，現在は多くのワープロに普通に備わっている機能のようであるが，印刷イメージの画面表示も欠かせないな。印刷する間待たされたうえ，紙の無駄までできるというのは，果てしなくプンスカだからな。

そいで，やっぱりコマンド体系はMicro Emacs系だな。基本的には[CTRL]-[～]と，[ESC]＋［～］で勝負するのである。で，まかり間違ってもキーボードにテンプレートをかぶせたくなるようなウニなコマンド体系にだけはしたくないと思うのである。

だいたい今の日本のワープロ/エディタはなんなんだ。Emacsの設計思想を多少なりとも理解しているのならば，あんなわけのわからないコマンド体系ができるわけがないと思うのだが。責任者出てこい。前にも書いたことがあるが「ファンクションキーは押しづらい」という当たり前のことを，いったいいつになったら理解してくれるのだろうか。

ほかにもいくらでもボヤくネタはあるが，とにかくキレの違うコマンド体系で勝負したいと思っているのである。そう，雷語1号は，パンチの効いたナウい日本語ワープロなのである。

FEPも一味違うぞ

FEPであるが，やっぱりASK68Kをそのまま使うというのも芸がない。そこで雷語1号のFEPは「雷太（サンダーボーイ）」である（香港製で「ドラゴン」なんて名前のマシンがあったらぜひ移植してみたいな)。

普通のFEPの基本動作といえば，変換モードに入ると画面の一番下の行が変換用に調達されて，その行で[変換]，［次候補］，［次候補］，［確定］とかをやるわけだな。

で，一見当然に思えるこのやり方が，おっとろしく古い基本設計のマシンにとらわれた，ジャップな方式であることにお気づきであろうか。そう，このような形式のFEPは，アメックスのダイレクトメールの中に入っている切手とボールペンのようにセセコマしいのだ。あんなものは，結局はキャラクタ用の画面を1面しか持ってなかった時代の盲腸なのである。

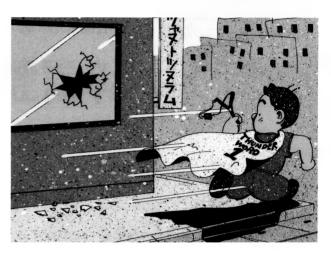
その点X68000は立派なビットマップ画面を持っている(しかも4プレーン)。そこでだな，FEPはそのうちの2プレーンを使用するのだ。はっきり言ってしまえば，マウスの右ボタンを押すと出てくる例の「仮想キーボード」と同じノリで「変換作業場」が出現するのである。で，実はそのFEPは常にWARMな状態で待機しているから，［CTRL］-［XF1］が押されたならば，電光石火で呼ばれて飛び出てジャジャジャジャーンなのである。当然，設定によって変換行は何行でもいいわけだな。よって最大の32行モードにして，パレットを不可視にして（この場合マウスカーソルも見えなくなるけど）おけば，たちまち，

スクリーンエディタが出現する

ではないかっ！

タネを明かせば，所詮画面切り替えにすぎないのであるが，使用感はマルチジョブである。もちろん機能などに多少の制限が付く予定であるが（メモリの占有量とかがあるからね)，基礎的な部分，たとえば編集機能のほとんど，ロード，セーブ，結合，印刷などはスパスパとできてしまうのである（もちろん，リッチに数メガを増設すれば，しっかりと雷語が100％動作する予定である)。これで，ちょっとした文章ならヒョヒョヒョイのンパンパとできてしまうのだ(おっと，書いててアルゴエディタを思い出したぜ)。こんなことも，CONFIG.SYSで，

DEVICE ＝ 3DERBOY.SYS #L32

と登録しておくだけでできてしまうのだっ。う～ん，なんて便利なんだろう。X68000であれば，これぐらいのものができてしまうということは，踏み台昇降の後では脈拍を数えなければいけないのと同じくらい確かなのだ。なんと使用者友好的であることよ。どだ？　どだ？

当然ながら，雷語の最初のものはX68000で発表されるわけであるが，特に温情を垂れて，雷語ダッシュとかゆーやつを他機種に移植してやってもよいと思っているが，挨拶に来る際には山吹色のモナカを忘れぬよーに。

なお，雷語1号はあくまで巨大な地殻変動の前兆にすぎないのである。すなわち，やがては雷語2号が出現し，さらには3号（女性用で「アマゾン」と発展していき，4号，5号，青の6号（ふ，古い)，……，28号で音声認識によるコントロールとなる予定である。文句あるか。

入手方法について

さて，この雷語1号の配布方法であるが，「フリーウェアにして，サポートは気が向いたらね」というのと，「ホントに売っちゃう。ちゃんとサポートやバージョンアップもすーるする」という2つの手があるわけだが，どっちにするかは，まあ，出来具合を見てから決めよう。本当のことを言うと，どこかのソフトハウスがちゃんとしたやつをさっさと作ってくれる（移植でもいいんだけど）のが一番ありがたいんだけどねぇ。それなりのレベルのものができれば，かなり売れるはずなんだけど。ぶつぶつ。

最後に念のため言っておく。こんなことを書いたからといって「まだできとらんのか！　さっさと出せ！」などと言ってせかさないよーに。そーゆー奴にはNHKの集金人を差し向けてやるからな。聞くところによると，あれは歩合制だそうだから，うかうかしてると尻小玉を抜かれるぜ。

ちなみに，雷語1号の開発の合言葉は「BTRONに続け！」である（笑)。

ASK68K用辞書メンテナンスツール《前編》

辞書整備基本編

Murata Toshiyuki
村田 敏幸

日本語入力の効率を上げるには辞書を鍛えることがいちばんです。今月と来月にわたってASK68Kの辞書を整理するためのツールをお届けします。日本語環境の整備も自分の手で行うことができるのです。

ここにASK68Kの辞書を整理するツールがある。なぜかある。とにかくある。辞書中の単語の一覧をテキストファイルに落とすプログラムに始まり，単語の一覧ファイルから新規に辞書を作成するもの，辞書中の単語をまとめて削るもの，辞書に単語をまとめて追加するもの，ほかの日本語フロントエンドプロセッサ用の辞書をASK用にコンバートする手助けをするもの，辞書が壊れていないかどうか検査してある程度修復するもの，これらに加えて大小の補助ツールが何本か……。ASKの辞書に関しては必要以上のことができるといっても過言ではないほどの充実したラインアップだ（うんうん）。

特集にかこつけて，今月と来月の2回でこれらのツール（全部は無理だが）を読者にお裾分けしようと思う。つまらないものですが，よろしければどうぞ，ってなもんだ。

でも変なんだよな。どうしてこんなツールを作ったんだろう？　しかも，こんなに充実させるほどムキになって。自分で作っておきながら，作成した動機がどうもはっきりしない。

作ったのはもうかなり前で，タイムスタンプを見ると去年の9月に最終的な修整を加えたことになっている。ますます変だ。当時の僕がこんなプログラムを必要とするはずがない。

そりゃあ，ASKの辞書はデカい。郵便番号辞書やら不要な固有名詞やらを削って，すっきりさせたいと思ってもなんの不思議もない。

そりゃあ，ASKの辞書はデカいわりには登録語が貧弱でJIS第2水準の熟語なんか全滅に近い。よその日本語フロントエンドプロセッサの辞書から単語を吸い上げてパワーアップできたらなーと夢見たとしてもバチはあたるまい。

そりゃあ，DICM.Xは使いにくい。誤変換によって積もり積もったサブ辞書中のゴミ（単漢字など）を削ろうと思い立ったとしようか。DICMを立ち上げ，メニューから単語削除を選び，問われるままに辞書ファイル名を入力する。と，ここで金縛りにあってしまう。読みを入力しなければ単語が削れない！　あらかじめ，削りたい単語を紙に書き出すか，プリンタに打ち出しておくかしなければ，どの単語を削ったらよいかがわからないとは。仮に削りたい単語をリストアップできたとしても，メモを見ながら読みを打ち込んでリターン，カーソルキーで単語を選んでリターン，よろしいですかの問いかけにYと答え，終わりますかの問いにNと答え，読み入力，単語選択，Y，N，読み入力，単語選択，Y，N……あっというまに嫌になる。そもそも，単語1個削るたびに“終わりますか？”って聞いてくるのはなんのつもりだ？

どれも立派な動機だと思ったかもしれない。確かにASKを頻繁に使う人にとってはそうだろう。ところが，僕は2年前から恥ずかしながら新松の人でね（ある日を境に突然WP.Xに耐えられなくなり，某国民機を買いに秋葉原へ走った）。ちょうど，去年の秋なんていったらWP.XはおろかASKを使う機会もほとんどなかったはずなのだ。なのにどうして，これだけの数のプログラムを作ったんだろう。謎だ。

ま，最近はX68000上で原稿以外のちょっとした文書（作ったプログラムの仕様書というか覚え書きみたいなのとか，いろいろ。あとで役に立つことは滅多にないが）なんかにASK68Kを使うことも多くなってきたから（WP.Xだけはどうしても駄目。絶対に駄目），いまごろになって，辞書を整理するプログラムを作っておいてよかったーと思っていたりする。備えあれば憂いなしとはよくいったものだ。

プログラムの内容

今月は基本セットを提供する。辞書内容をテキストファイルに落とす「DUMPDIC.X」，逆に単語の一覧ファイルから辞書ファイルを作成する「GENDIC.X」，単語一覧ファイルを食わせるとそのファイル中の単語を辞書からごっそり削除する「DELWORD.X」の3本だ。

あらかじめ断っておくが，これらのツールは扱いにくく得体のしれないバイナリファイルである辞書を扱いやすいテキストファイルに変換し，このテキストファイルのレベルで手を加えたうえで一括して辞書に戻す，という基本方針で作られている。いったんテキストに落としてしまえば，エディタで編集することもできれば，フィルタにかけることもできるわけだ。

各プログラムはいわゆるDOSコマンドっぽい体裁であり，COMMAND.X上で使うことを想定している。使用にあたっては，コマンドモードでの操作に習熟し，リダイレクトやパイプの利用方法を理解していることが望ましい。

また，ASKの辞書を自分なりに解析した結果と勘に基づいて作成したプログラムであり（内心ではある程度の自信を持っているとはいえ），残念ながら100%の動作保証をする勇気はない。ペーパーメディアで提供する以上，プログラム入力時に打ち込み間違いが生じることも考えられるし，万が一に備え，プログラムを走らせる前には辞書のバックアップを取っておくことをすすめる。

あと，打ち込んだプログラムは読者のものだ。改造しようが，ほかのプログラムに組み込もうが，友達にあげようが，好きにしてかまわない。この程度のプログラムでコピーライトを強硬に主張するほど酔狂ではない。ただし，ASKの辞書，X68K_M.DIC，X68K_S.DICは開発元のアクセス/シャープの著作物であり，ツールを使って加工したとしてもその事実に変わりはない。十分な配慮をもって取り扱ってほしい。

では各プログラムを紹介しよう。

●DUMPDIC.X

使用法：DUMPDIC ［スイッチ］ 辞書フ

ァイル名［”開始位置”［”終了位置”］］［/O出力ファイル名］

本ツール群の前提となるプログラムであり，なにはともあれ，まず必要。このプログラムがなければGENDIC.XやDELWORD.Xも意味をなさない。ただし，多少の制約つきでDICM.XにDUMPDIC.Xの代わりを務めさせることができる。具体的な手順はコラムにまとめておいた。

DUMPDICのもっとも単純な使い方は，

DUMPDIC X68K_S.DIC

のように，辞書ファイル名だけを指定した形式だ（拡張子の“.DIC”はつけてもつけなくてもよい）。この場合には，辞書の全内容が画面（標準出力）に書き出される。これを，

DUMPDIC X68K_S >file

のようにリダイレクトするか，

DUMPDIC X68K_S /Ofile

と，/Oスイッチに続けて出力ファイル名を指定すれば単語の一覧をファイルに落とすことができる。以下，こうやって作成したファイルのことを単語ファイルと呼ぶことにする。

サブ辞書はともかく，メイン辞書の内容を書き出すと1Mバイトを優に越える単語ファイルが作成される。そこで，非ハードディスクユーザーの場合は，“読み”の範囲を区切って2度以上に分割して出力する必要が生じる。読みの範囲は，たとえば次のように指定する。

DUMPDIC X68K_S ”あ” ”こんんん” >file1

この例では読みが“あ”～“こ”で始まる単語がfile1に出力される。なお，ここでは全角ひらがなで読みを指定しているが，DUMPDICは全角カタカナ，半角カタカナでの指定も受け付ける。また，

DUMPDIC X68K_S ”さ” >file2

のように，終了位置は省略してもよい。その場合は辞書の最後までとみなされる。開始位置を省略し，終了位置だけを指定することはできないが，開始位置に空文字列を指定して，

DUMPDIC X68K_S ”” ”こんんん”

とやれば同じ効果が得られる。

スイッチで指定することで特定の品詞だけを抽出することもできる。

/A	形容詞と形容動詞
/D	副詞
/N	固有名詞以外の名詞
/P	固有名詞
/T	単漢字
/V	動詞

当然，スイッチは複数指定してもよく，

DUMPDIC /N /P X68K_S

なら，辞書中のすべての名詞が出力される。また，

DUMPDIC /E /N /P X68K_S

のように/Eスイッチと併用すると，“指定の品詞以外”が対象になる。この例では名詞以外の全単語を出力している。

さらに，/Mスイッチを指定するとメニューから品詞を選ぶモードに入る。この方法を使えば“か行5段活用動詞のみ”といった細かな指定もできるようになる。メニューの操作は，

スペース	：選択・非選択の切り替え
リターン	：確定，実行
ESC	：中断

によって行う。DICM.Xのようにリダイレクトしたからといってメニューが表示されなくなるような間の抜けたことはない。

あと，おまけ的なスイッチがいくつかある。/Cスイッチをつけると単語を一覧表示する代わりに該当単語の個数のみを表示するようになる。/H，/G，/Kの各スイッチを指定すると“読み”をそれぞれ，

/H	半角カタカナ
/G	全角ひらがな
/K	全角カタカナ

で出力するようになる。/H，/G，/Kのいずれも指定されない場合は/Kとみなされる。/Hを指定すれば，作成される単語ファイルの大きさを若干小さくすることができるが，その代わりエディタなどでの編集は面倒になることが予想される。

DICM.XをDUMPDIC.Xの代わりに使う方法

リストを入力する手間を惜しむ読者のために，DICM.Xで単語ファイルを作成する方法を2つ紹介しておく。それぞれ一長一短があるから，うまく使い分けるとよいかもしれない。

その1：リダイレクトを利用する

安直な分，“メニューもリダイレクト先にいってしまうので画面になにも表示されなくなる”という大きな欠点がある（DICMが悪い）。具体的な手順は以下のとおり。

1）　まず，ふつうにDICMを起動し単語の一覧表示をしてDICMを終了するまでの一連のキー操作を覚えて（メモして）おく。

例）　下向きカーソルキーを2回押して，リターンキーを1回押して，それから～。

2）　いったんDICMを終了し，

A > DICM > B:WORDFILE

のように，標準出力を適当なファイルにリダイレクトして再起動する。画面にはなにも表示されないが気にしない。

3）　1）で覚えたとおりに，“単語一覧メニューを選び一覧を開始するまで”のキー操作を正確に再現する。途中で間違えたら，“インタラプトスイッチを押す”，“プリンタをオフライン状態にしてCOPYキーを押す”などの方法でDICMを強制終了し最初からやり直すのが手っ取り早い。

4）　ディスクが止まるのを待って，“単語一覧メニューを抜け，DICMを終了するまで”のキー操作を再現する。

5）　無事，COMMAND.Xのプロンプトが出たら出力ファイル（上の例ではWORDFILE）をエディタで読み込み，最初と最後についたゴミを削ってセーブし直せば完成。

その2：空の辞書との辞書差分を利用する

あまり使わない機能なので気づかなかったのだが，DICMの辞書差分メニューでは出力先にファイルを指定することができる。“X－0＝X”だから，空の辞書との差分をとることで単語の一覧ファイルが得られる。出力範囲を指定することができないのが難。

1）　単語がなにも登録されていない空の辞書を用意する。購入したときのままのX68K_S.DICがまさにそれ。運悪くシステムディスクが見当たらないような場合には，次のX_BASICプログラムで空の辞書を作ることもできる（要は8192バイト以上の00_Hだけからなるファイルがあればよい）。

```
10 dim char dmy(8191)
20 int fp
30 fp = fopen( "null.dic","c" )
40 fwrite( dmy, 8192, fp )
50 fclose( fp )
```

2）　DICM.Xを起動する。

3）　メニューから辞書差分を選び，辞書ファイルとして1）で用意した空の辞書，参照ファイルとして一覧表示したい辞書ファイル名，出力ファイルとして適当なファイル名を指定する。

4）　よろしいですかの問いかけにYと答えれば処理が始まる。

さて，DICM.Xを使って得られる単語ファイルはDUMPDIC.Xの出力形式とは異なる。が，GENDIC.XやDELWORD.Xはどちらの形式も受け付けるからその点では安心してもらってよい。

ただ，DUMPDIC.Xは各行に必ず“読み”をつけるのに対して，DICM.Xの場合は同一の読みが続く場合は読みが省略される。これは，単語ファイルの部分をFIND.Xなどを使って抽出する際などに問題になるかもしれない。たとえば，

```
ア：  阿    ／単漢字
イ：  行    ／カ五動詞
      位    ／単漢字
       ⋮
```

という単語ファイルから，FIND.Xを使って単漢字だけを選び出すとすると，

```
ア：  阿    ／単漢字
      位    ／単漢字
       ⋮
```

となってしまう。GENDIC.Xなどでは，読みがついていない行は直前の行と同じ読みとみなすから，これは，

```
ア：  阿    ／単漢字
ア：  位    ／単漢字
       ⋮
```

と解釈されるので，うまくないことがわかると思う。

●GENDIC.X

使用法：GENDIC ［/F空き容量］ ［単語ファイル］ ［/O］出力辞書ファイル

DUMPDICなどで作成した単語ファイルから新規に辞書ファイルを生成する。

GENDIC WORDFILE USER.DIC

もしくは，

GENDIC WORDFILE /OUSER.DIC

のようにして使うのが基本だ。上の例では単語ファイルWORDFILEから辞書ファイルUSER.DICを作成している。/Oスイッチは単にほかのプログラムとの対称性を保つために用意されているだけで，通常は省略できる。また，例によって拡張子“.DIC”はつけなくてもよい。

あまりやらないことだろうが，パイプでつなぐような用途も想定して，単語ファイルの指定が省略された場合は標準入力から入力するようにしてある。この場合は/Oスイッチをつけて，パラメータが単語ファイルではなく辞書ファイル名であることを明示する必要がある。

DUMPDIC /N X68K_S | GENDIC /OUSER

特に指定されない場合，GENDICは単語を可能な限り詰め込んで隙間のない辞書を作成する。あとから単語登録ができるようにゆとりを設けたい場合は，スイッチ/Fにより空き容量の割合（DICMで辞書再編成をする場合と同様，単位は%）を決める。

GENDIC /F20 WORDFILE USER

空き容量は0から最大50%まで許すようにしてある。

GENDICに与える単語ファイルのフォーマットはDUMPDICの出力形式と同じで，1行に1単語分，

読み　登録語　品詞名

の順序で並んだものだ。各フィールドは半角スペースまたはタブで区切る。読みは全角・半角のいずれでも構わないが，必ず行頭から（空白を入れずに）書き始めなければならない。行頭が空白の場合は直前の行の読みを引き継ぐ。要するに，

```
ア    阿    単漢字
      亜    単漢字
      唖    単漢字
      ⋮
```

は，

```
ア    阿    単漢字
ア    亜    単漢字
ア    唖    単漢字
      ⋮
```

とみなされる。

また，一応，

ア：　阿　　/単漢字

のようなDICM.Xの単語一覧の出力形式も受け付けるように作ってある。

ときにGENDICは単語ファイルの行番号とともに以下のような警告メッセージを出す場合がある。

・無効行がありました

単語ファイルの形式が正しくない。3つのフィールドが揃っていないと思われる。

・無効な品詞名［××］がありました

言葉どおり。DUMPDICでサブ辞書の全内容を出力し，GENDICに与えたときにもこの警告が出る場合がある。これはDUMPDICとGENDICの仕様であって，害はないから無視して構わない。ASKは一部の動詞を内部で特別扱いしており，辞書学習によりこれがサブ辞書ににじみ出てくることがある。この部分は標準の辞書フォーマットから若干はずれた形式になっており，GENDICはこれに対応していない（必要もない）。そこで，DUMPDICはこれら特殊な単語の品詞に“＊”をつけて出力することによって，故意にGENDICの処理からはずすようにしてあるのだ。

・読み［××］に対応する単語が1ページに収まりませんでした

ASKの辞書は1024バイトのページに分割されていて，同一の読みの単語が複数ページにまたがることを許していない。GENDICは1ページに収まる範囲のみを出力し，残りは切り捨てる。

・読み［××］に対応する単語が100個を越えました

理由はわからないし，そもそも理由なんてないのだろうが，ASKには“ひとつの読みに対する登録語は100個まで”という変な制約があり，それを越える単語は辞書中に存在してもかな漢字変換時に参照されない。GENDICはASKの将来的なバージョンアップに期待して，一応100個を越える単語も辞書に登録する。ところで，X68K_M.DIC中，読み「こう」には103語が登録されている。困ったものだ。

・二重登録がありました

言葉どおり。GENDICは単にこれを無視する。

・単語の並び順が正しくありません

これはGENDICの自慢できない仕様だ。GENDICに食わせる単語ファイルはあらかじめ“ASKの辞書中における読みの内部コード順にソートされてなければならない”ということになっている。DUMPDICの出力をそのまま使う分には問題は生じないが，単語ファイルをエディタなどでゼロから作成する場合には注意が必要だ。来月，専用のソートプログラムを提供する予定でいるが，暫定的な対処方法を示しておこう。単語の並び順の乱れを検出したとき，GENDICは警告メッセージと同時に標準出力に登録できなかった単語ファイルの該当行を出力する（警告メッセージは標準エラー出力に書き出している）。ここで，

GENDIC WORDFILE USER >ERR

のようにリダイレクトしておくと，“登録しそこねた単語だけからなる単語ファイル”が得られることになるから，この単語ファイルを使って，

GENDIC ERR USER1

のようにして別の辞書ファイルを作り，DICMで併合すればよい。

●DELWORD.X

使用法：DELWORD ［スイッチ］ ［単語ファイル名］ ［/O］辞書ファイル名

単語ファイルで指定した単語を辞書ファイルからまとめて削除する。単語ファイル名が省略された場合は標準入力から入力するあたりはGENDICと同様だ。/E，/V 2つのスイッチがある。/Eが指定されると，単語ファイル中の無効行をエラーファイルDELWORD.ERRに書き出す。また，/Vが指定されると実行経過を表示する。このとき，無効行は黄色で，元々辞書中になかった単語は青で標準エラー出力に書き出し，削除できた単語は“標準出力”に出力する。

DELWORD /V WORDFILE USER >UNDO

のようにリダイレクトしておけば，“削除した単語だけからなる単語ファイル”が得られる。

●SPLIT.X

使用法：SPLIT ［スイッチ］［入力ファイル名］［［/O］出力ディレクトリ］

オマケ。辞書を整理する過程では結構大きなファイルを扱うことになり，場合によってはエディタで読み込めなかったり，ハードディスクからフロッピーディスクに転送できなかったりすることが考えられる。SPLITはテキストファイルをいくつかの小さなファイルに分割する小ツールだ。

出力ファイルは入力ファイル名の拡張子を“.000”，“.001”，……というように数字で置き換えたファイル名で，特に指定されない限りカレントディレクトリ上に作

られる。分割する大きさは，/Bか/Lスイッチで指定する。それぞれ後ろに数字を伴い，/Bスイッチの場合はバイト単位，/Lスイッチの場合は行単位で分割する。どちらのスイッチも指定されない場合は/L1000とみなされる。

応用例

●メイン辞書から郵便番号辞書を削る

1)　DUMPDICで郵便番号辞書を抜き出す。

　　DUMPDIC /M X68K_M "000""999" >YUBIN

のようにしてDUMPDICを起動し，メニューから品詞は“地名”を選択する。

2)　抜き出した単語ファイルYUBINをDELWORDにかける。

　　DELWORD YUBIN X68K_M

3)　このままでは辞書ファイルの大きさは変わらないので，DICMで辞書を再編成する。

●独立した郵便番号辞書を作る

1)　上の1)と同様の方法で郵便番号辞書の単語ファイルを作成する。

2)　これをGENDICにかける。

　　GENDIC YUBIN YUBIN

●多くの単語をまとめてメイン辞書に追加登録する

1)　登録したい単語だけからなる単語ファイルをエディタなどで作成する。このファイルを仮にWORDFILEとする。

2)　WORDFILEをGENDICにかける。

　　GENDIC WORDFILE TEMP

3)　2)で作成した辞書をDICMでメイン辞書と併合する。

入力・コンパイル方法

　プログラムはごくわずかな部分を除き，Cで記述してある。各プログラムに共通で使える関数が多くあることから，ソースは複数に分割されており，リンク時に結合するようになっている。各プログラムごとに必要なファイルを表1にまとめておくから，それぞれエディタで入力し，指定されたとおりのファイル名でセーブしておいてもらいたい。

　リスト中，左端の行番号は入力する必要がない。また，全角文字と半角文字の違いは重要な意味を持つから，注意を払ってほしい。

　入力が済んだらコンパイルにとりかかる。一気に全部まとめてコンパイルし，実行ファイルを作成することもできるが，1カ所エラーが出て修正するたびに全ファイルをコンパイルし直すのは合理的ではないので，個別にコンパイルし，そののちリンクするという段階を踏むのがよいだろう。MAKEを利用できる環境にあればベストなのだが，純正品のMAKEはXC Ver.2.0を待たなければならない。

　リストは拡張子によって，3種類に分類され，それぞれ対応が違う。

・拡張子が“.H”のもの

　これはヘッダファイルで，コンパイル時に自動的に参照されるから，単にカレントディレクトリに置いておけばよい。

・拡張子が“.S”のもの

　VFPRINTF.Sだけがこれに該当する。アセンブリソースだから，

　　AS VFPRINT

によってアセンブルする。なお，このソース中ではvfprintfというANSI Cで定義された関数のコンパチ品が定義されている。XC Ver.2.0には同関数が用意されるのは間違いないから，Ver.2.0が手に入ればすぐに不要になるはずのものだ。また，VFPRINTF.S中にはどういうわけかprintfとfprintfも再定義されている。XC Ver.1のprintfはエラーを返さないという古い仕様なので，エラーを返す版を用意した。これもXC Ver.2.0が出るまでの暫定的な処置だ。

・拡張子が“.C”のもの

　ほとんどがこれ。XCを使ってコンパイルするときには，次のようにする。

　　CC /L DUMPDIC.C

　ここでLは大文字でなければならない。CC.Xはコンパイル過程でできたアセンブリソースを消去してくれないので，作業ディスクの容量が不足気味であれば，コンパイル後に，

　　DEL DUMPDIC.S

によって手作業で消そう。

　GCCを利用するときには多少問題がある。移植の版にもよるのだが，少なくとも本誌6月号の付録ディスクに収録されたGCC V1.36.01は日本語に対応していないので，ふつうにコンパイルしたのではプログラム中のメッセージの一部が化けてしまうのだ。これを避けるためには，付録ディスクに収録されているKNJ2OCT.Xを使ってソース中の全角文字を8進エスケープシーケンスに変換してやる必要がある。ソース1本ごとに，

　　KNJ2OCT DUMPDIC.C _DUMPDIC.C

とやって，生成された_DUMPDIC.CをGCCでコンパイルする。GCCの場合，リンク直前で止めるには，

　　GCC -c _DUMPDIC.C

のように-cスイッチ（cは小文字）を使う。好みによっては最適化オプションをつけて，

　　GCC -c -O -fstrength-reduce -fomit-frame-pointer-finline-functions _DUMPDIC.C

とでもやればよい。リスト1のようなバッチファイルを作れば楽だろう。

　以上の作業によって，カレントディレクトリにはソースファイルの拡張子が“.O”に置き換えられたファイルがたくさんできているはずだ。これらをリンクすれば実行ファイルが得られる。XC，GCCのどちらでコンパイルした場合もリンクにはCC.Xを使うのが楽だろう。DUMPDIC.Xを作成するのなら次のようになる。

　　CC /Y DUMPDIC.O PAGENO.O WCLASS.O SELCLASS.O STRFUNC.O MISC.O VFPRINTF.O

　どのプログラムもDOSLIB.A，IOCSLIB.Aを使用しているので，忘れずに/Yスイッチをつける（Yは大文字）。おっと，GCCでコンパイルした場合にはKNJ2OCTを通したときにファイル名を変えてしまっているから次のようにリンクする。

　　CC /Y /ZDUMPDIC.X _DUMPD

表1

	dumpdic.x	gendic.x	delword.x	split.x
dumpdic.c	○			
gendic.c		○		
delword.c			○	
split.c				○
parseline.c		○	○	
pageno.c	○		○	
selclass.c	○			
wclass.c	○	○	○	
strfunc.c	○	○	○	
misc.c	○	○	○	○
vfprintf.s	○	○	○	○
mydef.h	○	○	○	○
misc.h	○	○	○	○
strfunc.h	○	○	○	
myerror.h	○	○	○	○
dictools.h	○	○	○	

リスト1　KGC.BAT

```
knj2oct.x %1.c _%1.c
gcc -c -O -fstrength-reduce -fomit-frame-pointer -finline-functions _%1.c
```

IC.O _PAGENO.O …

最後になったが，掲載したリストの分割の仕方は大雑把であり，使っていない関数も実行ファイルに含まれてしまっている。関数1個ごとに別ファイルにしておけばよかったのだが，さすがに20～30個のオブジェクトをリンクしてください，とはいいにくかった。どうしても無駄なコードを削りたいという人は，各自対応してもらいたい。

＊　　＊　　＊

というあたりで来月へと続く。来月は辞書の単語ファイルレベルでのコンバータと，ワンキーでメイン辞書を切り替える常駐プログラム，本誌に以前掲載されたものよりも若干情報量の増えたASKの辞書構造解析結果などが予定されている。

辞書のコンバータに関しては，PC-9801用のメジャーどころからのコンバートをひととおりサポートするべく現在拡張作業を進めている。コンバータとはいっても，あくまで単語ファイルのフォーマット変換をするフィルタであり，過度な期待は遠慮願う。また，PC-9801用でもっとも広まっている某ワープロ用の日本語入力フロントエンドプロセッサからのコンバートはサポートするつもりがない。理由は，嫌いだから，だ。あらかじめご了承願う。

リスト2 DUMPDIC.C

```
  1: /*
  2:         DUMPDIC.X
  3:         ASK68Kの辞書ファイル内容を表示する
  4:
  5:         dumpdic.c pageno.c selclass.c
  6:         wclass.c strfunc.c misc.c
  7:         vfprintf.s
  8:         mydef.h misc.h strfunc.h myerror.h dictools.h
  9:         doslib.a iocslib.a
 10: */
 11:
 12: #include    "mydef.h"
 13: #include    <stdio.h>
 14: #include    <stdlib.h>
 15: #include    <ctype.h>
 16: #include    <string.h>
 17: #include    <limits.h>
 18: #include    "dictools.h"
 19: #include    "misc.h"
 20: #include    "strfunc.h"
 21: #include    "myerror.h"
 22:
 23: PACKEDSTR progname = "DUMPDIC";
 24: PACKEDSTR usagemes =
 25: "機　能：ASK68Kの辞書ファイル内容を表示します¥n¥
 26: 使用法：%s [スイッチ] 辞書ファイル[.DIC] ¥
 27: [¥"開始位置¥" [¥"終了位置¥"]] [/O出力ファイル]¥n¥
 28: ¥t/V¥t動詞を対象にする¥n¥
 29: ¥t/A¥t形容詞/形容動詞を対象にする¥n¥
 30: ¥t/D¥t副詞を対象にする¥n¥
 31: ¥t/N¥t名詞（固有名詞を除く）を対象にする¥n¥
 32: ¥t/P¥t固有名詞を対象にする¥n¥
 33: ¥t/T¥t単漢字を対象にする¥n¥
 34: ¥n¥
 35: ¥t/E¥tスイッチで指定した品詞以外を対象にする¥n¥
 36: ¥t/M¥tメニューによって対象品詞を選択する¥n¥
 37: ¥n¥
 38: ¥t/C¥t該当単語数のみを表示する¥n¥
 39: ¥t/H¥t見出し語を半角カタカナで出力する¥n¥
 40: ¥t/G¥t見出し語を全角ひらがなで出力する¥n¥
 41: ¥t/K¥t見出し語を全角カタカナで出力する（デフォルト）¥n¥
 42: ¥n¥
 43: ¥t対象品詞が指定されない場合は全品詞を対象にする¥n";
 44:
 45: #define     DONE        1
 46: #define     MORE        ( !DONE )
 47:
 48: static boolean countonly;
 49: static boolean hankaku;
 50: static boolean hiragana;
 51: static unsigned int flags;
 52: static int maxpage, pageno;
 53: static int wordctr = 0;
 54:
 55: static void warn()
 56: {
 57:     fprintf( stderr,
 58:     "警告：辞書の%dページ目に異常が見られます¥n", pageno + 1 );
 59: }
 60:
 61: static int dumprec( fp, buff, st, ed )
 62: FILE *fp;
 63: STRPTR buff, st, ed;
 64: {
 65:     extern STRPTR classtbl[];
 66:     STRPTR p, q, buffe;
 67:     STR index0, index, word;
 68:     unsigned int classcode;
 69:     int len;
 70:     boolean spec, wflg = FALSE;
 71:
 72:     for ( buffe = buff + PAGELEN; ( q = buff ) < buffe; ) {
 73:         len = *q++;
 74:         len += ( ( int ) *q++ ) << 8;
 75:         if ( len == 0 )
 76:             return ( MORE );
 77:         buff += len;
 78:         len = *q++;
 79:
 80:         if ( buff > buffe || len == 0 ) {
 81:             warn();
 82:             break;
 83:         }
 84:
 85:         memcpy( index0, q, len );
 86:         *( index0 + len ) = '¥0';
 87:         q += len;
 88:
 89:         if ( strcmp( index0, ed ) > 0 )
 90:             return ( DONE );
 91:         if ( strcmp( index0, st ) >= 0 ) {
```

```
 92:             setindex( index0,
 93:             ASKtoSJIS( index, index0 ), hankaku, hiragana );
 94:
 95:             for ( ; q < buff; ) {
 96:                 classcode = *q++;
 97:                 for ( spec = FALSE; *q <= 0x1f; q++, spec = TRUE
    );
 98:
 99:                 for ( p = word; q < buff && *q >= 0x20; )
100:                     *p++ = *q++;
101:                 *p = '¥0';
102:
103:                 if ( classcode == 0 || classcode >= 0x1f ) {
104:                     if ( !wflg )
105:                         warn();
106:                     wflg = TRUE;
107:                 } else if ( flags & ( 1 << classcode ) ) {
108:                     if ( !strcmp( "@", word ) )
109:                         strcpy( word, index );
110:                     settab( word, 8 * 3 );
111:
112:                     if ( !countonly
113:                     && fprintf( fp, "%s%s%s%s¥n",
114:                     index0, word, classtbl[ classcode ],
115:                     ( spec ) ? "*" : "" ) == EOF )
116:                     diskfull( fp );
117:                     wordctr++;
118:                 }
119:             }
120:         }
121:     }
122:     return ( MORE );
123: }
124:
125: void main( argc, argv )
126: int argc;
127: STRPTR *argv;
128: {
129:     PAGEBUFF pagebuff;
130:     STR st, ed;
131:     STRPTR dicfile = NULL;
132:     STRPTR destfile = NULL;
133:     boolean except = FALSE;
134:     boolean menumode = FALSE;
135:     FILE *sourfp, *destfp;
136:
137:     int KEYSNS( void );
138:     unsigned int selclass( unsigned int );
139:     HEADBUFF *readheader( FILE * );
140:     int getpageno( HEADBUFF *, STRPTR );
141:     int getmaxpageno( HEADBUFF * );
142:     void seekpage( FILE *, int );
143:
144:     for ( *st = *ed = '¥0', flags = 0; --argc; ) {
145:         if ( **++argv != '¥0'
146:         && strchr( SWITCH_DELIMITER, **argv ) != NULL ) {
147:             unsigned int sc;
148:             sc = *++*argv;
149:             switch ( tolower( sc ) ) {
150:                 case 'a':
151:                     flags |= ( 3 << 13 );
152:                     break;
153:                 case 'd':
154:                     flags |= ( 1 << 29 );
155:                     break;
156:                 case 't':
157:                     flags |= ( 1 << 18 );
158:                     break;
159:                 case 'n':
160:                     flags |= ( 7 << 15 );
161:                     break;
162:                 case 'p':
163:                     flags |= ( 31 << 19 );
164:                     break;
165:                 case 'v':
166:                     flags |= 0x1ffe;
167:                     break;
168:                 case 'e':
169:                     setflag( &except );
170:                     break;
171:                 case 'm':
172:                     setflag( &menumode );
173:                     break;
174:                 case 'c':
175:                     setflag( &countonly );
176:                     break;
177:                 case 'h':
178:                     setflag( &hankaku );
179:                     break;
180:                 case 'g':
181:                     setflag( &hiragana );
```

```
182:                    break;
183:                case 'k':
184:                    if ( hiragana | hankaku )
185:                        usage();
186:                    break;
187:                case 'o':
188:                    setptr( &destfile, ++*argv );
189:                    break;
190:                default:
191:                    usage();
192:                    break;
193:                }
194:            } else if ( dicfile == NULL ) {
195:                dicfile = chkext( *argv, ".DIC" );
196:            } else if ( *st == '¥0' ) {
197:                SJIStoASK( st, strzenkata( st, *argv ) );
198:            } else if ( *ed == '¥0' ) {
199:                SJIStoASK( ed, strzenkata( ed, *argv ) );
200:            } else {
201:                usage();
202:            }
203:        }
204:
205:        if ( dicfile == NULL )
206:            usage();
207:
208:        if ( ( sourfp = fchkopen( dicfile, "rb" ) ) == NULL )
209:            fatal_error(
210:            "辞書ファイル[%s]がみつかりません", dicfile );
211:
212:        if ( *ed == '¥0' )
213:            *ed = UCHAR_MAX;
214:        if ( strcmp( st, ed ) > 0 ) {
215:            STR temp;
216:            strcpy( temp, st );
217:            strcpy( st, ed );
218:            strcpy( ed, temp );
219:        }
220:
221:        if ( flags == 0 && !menumode )
222:            flags = UINT_MAX;
223:        else if ( except )
224:            flags = ~flags;
225:
226:        destfp = setstdout( destfile );
227:        resetstdin();
228:        breakset( NULL );
229:
230:        if ( !menumode || ( flags = selclass( flags ) ) != 0 ) {
231:            HEADBUFF *hp;
232:
233:            hp = readheader( sourfp );
234:            pageno = getpageno( hp, st );
235:            maxpage = getmaxpageno( hp );
236:            seekpage( sourfp, pageno );
237:            free( hp );
238:
239:            for ( wordctr = 0; pageno < maxpage; pageno++ ) {
240:                KEYSNS();
241:                if ( fread( pagebuff, 1, sizeof( pagebuff ), sourfp
)
242:                        != sizeof( pagebuff ) )
243:                    fatal_error( RERRORMES );
244:                else if ( dumprec( destfp, pagebuff, st, ed ) == DO
NE )
245:                    break;
246:            }
247:            if ( countonly ) {
248:                if ( fprintf( destfp, "該当単語数:%6d¥n", wordctr )
== EOF )
249:                    diskfull( destfp );
250:            } else if ( wordctr ) {
251:                fprintf( stderr, "%d個の単語がありました¥n", wordct
r );
252:            } else {
253:                fputs( "該当する単語はありませんでした¥n", stderr );
254:            }
255:        }
256:        fclose( sourfp );
257:        fclose( destfp );
258:
259:        exit( EXIT_SUCCESS );
260: }
```

リスト3 GENDIC.C

```
  1: /*
  2:         GENDIC.X
  3:         単語ファイルから新規に辞書ファイルを作成する
  4:
  5:         gendic.c parseline.c wclass.c
  6:         strfunc.c misc.c vfprintf.s
  7:         mydef.h misc.h strfunc.h myerror.h
  8:         doslib.a iocslib.a
  9: */
 10:
 11: #include    "mydef.h"
 12: #include    <stdio.h>
 13: #include    <stdlib.h>
 14: #include    <ctype.h>
 15: #include    <string.h>
 16: #include    "dictools.h"
 17: #include    "misc.h"
 18: #include    "strfunc.h"
 19: #include    "myerror.h"
 20:
 21: typedef enum {
 22:     OK, INVLIN, ILLCLASS, BUFFFLOW,
 23:     TOOMANY, DUPW, ILLORD,
 24: } STAT;
 25:
 26: PACKEDSTR progname = "GENDIC";
 27: PACKEDSTR usagemes =
 28: "機  能:ASK68Kの辞書ファイルを新規作成します¥n¥
 29: 使用法:MAKEDIC [スイッチ] [単語ファイル] [/O]辞書ファイル[.DI
   C]¥n¥
 30: ¥t/Fn¥t空き容量の設定 (n≦50%) ¥n";
 31:
 32: #define     NWORD       ( PAGELEN / 2 )
 33:
 34: static STR linbuf;
 35: static STRPTR wordfile = NULL;
 36: static STRPTR dicfile = NULL;
 37: static int linctr = 0;
 38: static int indexctr = 0;
 39: static HEADBUFF header;
 40: static STR wordbuf[ NWORD ];
 41: static unsigned char classbuf[ NWORD ];
 42: static int size = 0;
 43: static PAGEBUFF recbuff, pagebuff;
 44: static FILE *sourfp, *destfp;
 45: static int freesize = 0;
 46:
 47: static void warn( wno, lin, str )
 48: STAT wno;
 49: int lin;
 50: STRPTR str;
 51: {
 52:     static char *warnmes[] = {
 53:         NULL,
 54:         "無効行がありました¥n%s",
 55:         "無効な品詞名[%s]がありました¥n",
 56:         "読み[%s]に対応する単語が1ページに収まりませんでした¥n
",
 57:         "読み[%s]に対応する単語が100個を越えました¥n",
 58:         "二重登録[%s]がありました¥n",
 59:         "単語の並び順が正しくありません¥n",
 60:     };
 61:
 62:     fprintf( stderr, "%6d:¥t【警告】", lin );
 63:     fprintf( stderr, warnmes[ wno ], str );
 64:     if ( wno == ILLORD )
 65:         fputs( str, stdout );
 66: }
 67:
 68: static void putheader()
 69: {
 70:     if ( fseek( destfp, 0, SEEK_SET )
 71:     || fwrite( &header, 1, sizeof( header ), destfp )
 72:                             != sizeof( header ) )
 73:         diskfull( destfp );
 74: }
 75:
 76: static void putpage()
 77: {
 78:     int i;
 79:     STRPTR p;
 80:
 81:     if ( indexctr < MAXINDEX ) {
 82:         if ( fwrite( pagebuff, 1, PAGELEN, destfp ) != PAGELEN
) {
 83:             putheader();
 84:             diskfull( destfp );
 85:         }
 86:         p = pagebuff + 2;
 87:         i = *p++;
 88:         if ( i > 8 )
 89:             i = 8;
 90:         memcpy( header.index[ indexctr++ ], p, i );
 91:         memset( pagebuff, '¥0', PAGELEN );
 92:         size = 0;
 93:     } else {
 94:         putheader();
 95:         fatal_error( "辞書ファイルの大きさが限界を越えました"
);
 96:     }
 97: }
 98:
 99: static STAT putrec( index, ctr, f )
100: STRPTR index;
101: int ctr, f;
102: {
103:     STRPTR p = recbuff + 2;
104:     int len, mlen, tlen;
105:     int i;
106:     STAT stat = OK;
107:
108:     mlen = strlen( index );
109:     header.flag[ *index ] = TRUE;
110:     len = 2 + 1 + mlen;
111:     *p++ = mlen;
112:     strcpy( p, index );
113:     p += mlen;
114:
115:     for ( i = 0; i < ctr; i++ ) {
116:         tlen = strlen( wordbuf[ i ] );
117:         if ( len + tlen > PAGELEN ) {
118:             stat = BUFFFLOW;
119:         } else {
120:             *p++ = classbuf[ i ];
121:             strcpy( p, wordbuf[ i ] );
122:             len += tlen + 1;
```

```
123:            p += tlen;
124:        }
125:    }
126:    p = recbuff;
127:    *p++ = len & 0xff;
128:    *p++ = ( len >> 8 ) & 0xff;
129:    if ( size + len > PAGELEN - freesize )
130:        putpage();
131:    memcpy( pagebuff + size, recbuff, len );
132:    size += len;
133:    if ( f )
134:        putpage();
135:
136:    return ( stat );
137: }
138:
139: static void gendic( index, word, class )
140: STRPTR index, word, class;
141: {
142:     static STR index0;
143:     static int ctr = 0, toplin;
144:     STRPTR word0 = word;
145:     int class_code;
146:     int getclasscode( STRPTR );
147:
148:     if ( index == NULL ) {
149:         if ( ctr && putrec( index0, ctr, TRUE ) )
150:             warn( BUFFFLOW, toplin, index );
151:     } else {
152:         if ( !strcmp( index, word0 ) )
153:             word0 = ( STRPTR ) "@";
154:         if ( !( class_code = getclasscode( class ) ) ) {
155:             warn( ILLCLASS, linctr, class );
156:         } else if ( *index0 == '¥0' ) {
157:             toplin = linctr;
158:             SJIStoASK( index0, index );
159:             strcpy( wordbuf[ ctr = 0 ], word0 );
160:             classbuf[ ctr++ ] = class_code;
161:         } else {
162:             STR temp;
163:             int cmpstat, i;
164:
165:             SJIStoASK( temp, index );
166:             if ( !( cmpstat = strcmp( temp, index0 ) ) ) {
167:                 if ( ctr < NWORD ) {
168:                     if ( ctr == 100 )
169:                         warn( TOOMANY, toplin, index );
170:                     for ( i = 0; i < ctr; i++ ) {
171:                         if ( classbuf[ i ] == class_code
172:                         && !strcmp( wordbuf[ i ], word0 ) ) {
173:                             warn( DUPW, linctr, word );
174:                             break;
175:                         }
176:                     }
177:                     if ( i == ctr ) {
178:                         strcpy( wordbuf[ ctr ], word0 );
179:                         classbuf[ ctr++ ] = class_code;
180:                     }
181:                 }
182:             } else if ( cmpstat > 0 ) {
183:                 if ( putrec( index0, ctr, FALSE ) )
184:                     warn( BUFFFLOW, toplin, index );
185:                 toplin = linctr + 1;
186:                 strcpy( index0, temp );
187:                 strcpy( wordbuf[ ctr = 0 ], word0 );
188:                 classbuf[ ctr++ ] = class_code;
189:             } else {
190:                 warn( ILLORD, linctr, linbuf );
191:             }
192:         }
193:     }
194: }
195:
196: void main( argc, argv )
197: int argc;
198: STRPTR *argv;
199: {
200:     STR index, word, class;
201:     boolean dflt = TRUE;
202:     int KEYSNS( void );
203:
204:     while ( --argc ) {
205:         if ( strchr( SWITCH_DELIMITER, **++argv ) != NULL ) {
206:             unsigned int sc;
207:             sc = *++*argv;
208:             switch ( tolower( sc ) ) {
209:                 case '¥0':
210:                     setptr( &wordfile, stdin ); /*dummy*/
211:                     break;
212:                 case 'f':
213:                     setvalue( &freesize, ++*argv, 0, 50 );
214:                     break;
215:                 case 'o':
216:                     setptr( &dicfile, ++*argv );
217:                     break;
218:                 default:
219:                     usage();
220:                     break;
221:             }
222:         } else if ( wordfile == NULL ) {
223:             dflt = FALSE;
224:             wordfile = *argv;
225:         } else if ( dicfile == NULL ) {
226:             dicfile = *argv;
227:         } else {
228:             usage();
229:         }
230:     }
231:     if ( dicfile == NULL )
232:         usage();
233:     else
234:         dicfile = chkext( dicfile, ".DIC" );
235:
236:     resetstdin();
237:     if ( dflt ) {
238:         sourfp = stdin;
239:         wordfile = "[標準入力]";
240:     } else if ( ( sourfp = fchkopen( wordfile, "r" ) ) == NUL
    L ) {
241:         fatal_error( ROPENERRMES, wordfile );
242:     }
243:
244:     if ( ( destfp = fchkopen( dicfile, "wb" ) ) == NULL )
245:         fatal_error(
246:         "辞書ファイル[%s]が作成できません", dicfile );
247:
248:     breakset( NULL );
249:
250:     fprintf( stderr,
251:         "登録単語ファイル     :%s¥n登録対象辞書ファイル:%s¥n
     ",
252:         wordfile, dicfile );
253:
254:     if (fwrite( &header, 1, sizeof( header ), destfp ) != siz
    eof(header))
255:         diskfull( destfp );
256:
257:     freesize = ( freesize == 0 ) ? 2 : freesize * 10;
258:
259:     for ( *index = '¥0', linctr = 1;
260:         fgets( linbuf, NCHAR, sourfp  ) != NULL; linctr++ ) {
261:         KEYSNS();
262:         if ( parseline( linbuf, index, word, class ) )
263:             gendic( index, word, class );
264:         else if ( *linbuf != '¥0' )
265:             warn( INVLIN, linctr, linbuf );
266:     }
267:     gendic( NULL );
268:
269:     putheader();
270:
271:     fclose( sourfp );
272:     fclose( destfp );
273: }
```

リスト4 DELWORD.C

```
 1: /*
 2:         DELWORD.X
 3:         辞書ファイルから単語をまとめて削除する
 4:
 5:         delword.c pageno.c parseline.c wclass.c
 6:         strfunc.c misc.c vfprintf.s
 7:         mydef.h misc.h strfunc.h myerror.h dictools.h
 8:         doslib.a iocslib.a
 9: */
10:
11: #include    "mydef.h"
12: #include    <stdio.h>
13: #include    <stdlib.h>
14: #include    <ctype.h>
15: #include    <string.h>
16: #include    "dictools.h"
17: #include    "misc.h"
18: #include    "strfunc.h"
19: #include    "myerror.h"
20:
21: #define     NORMCOLOR    33
22: #define     SKIPCOLOR    31
23: #define     ERRCOLOR     32
24:
25: PACKEDSTR progname = "DELWORD";
26: PACKEDSTR usagemes =
27: "機  能:単語ファイルで指定された単語を¥
28: ASK68Kの辞書ファイルからまとめて削除します¥n¥
29: 使用法:%s [スイッチ] [単語ファイル] [/O]辞書ファイル[.DIC]¥n

30: ¥t/E¥t削除できなかった単語をDELWORD.ERRに書き出します¥n¥
31: ¥t/V¥t実行経過を報告します¥n";
32:
33: static HEADBUFF *hp;
34: static PAGEBUFF pagebuff;
35: static STR linbuf;
36: static STRPTR wordfile, dicfile;
37: static FILE *sourfp, *dicfp, *errfp;
38: static boolean report = FALSE;
39: static boolean verbose = FALSE;
40: static int invalidline = 0;
41: static int validline = 0;
42: static int linctr = 0;
43: static int maxpageno;
44: static int curpageno = -1;
45: static PACKEDSTR errorfile = "DELWORD.ERR";
46:
47: #define verboseprintf   if ( verbose ) fprintf
```

▶先日，武田信玄史跡巡りの旅に行ってきました。もう感動！ 勝頼の無念さがひしひしと伝わって涙が……。
谷口博一(24)大阪府

```c
48:
49: typedef enum {
50:     DONE, NOTFOUND, ERR
51: } STAT;
52:
53: static STRPTR searchrec( hp, buff, index )
54: HEADBUFF *hp;
55: STRPTR buff, index;
56: {
57:     int cmpstat;
58:     STR index0;
59:     int pageno, len;
60:     STRPTR p, q, r;
61:     void seekpage( FILE *, int );
62:
63:     for ( pageno = getpageno( hp, index );
64:         pageno < maxpageno; pageno++ ) {
65:         if ( curpageno != pageno ) {
66:             seekpage( dicfp, curpageno = pageno );
67:             if ( fread( buff, 1, PAGELEN, dicfp ) != PAGELEN )
68:                 fatal_error( RERRORMES );
69:         }
70:         for ( p = buff; ; p += len ) {
71:             len = p[0] | ( ( ( int ) p[1] ) << 8 );
72:             if ( len == 0 )
73:                 break;
74:             memcpy( index0, p + 3, p[2] );
75:             *( index0 + p[2] ) = '¥0';
76:             if ( ( cmpstat = strcmp( index0, index ) ) == 0 )
77:                 return ( p );
78:             else if ( cmpstat > 0 )
79:                 return ( NULL );
80:         }
81:     }
82:     return ( NULL );
83: }
84:
85: static STRPTR searchword( rec, index, word, classcode, reclen
 )
86: STRPTR rec, index, word;
87: unsigned int classcode;
88: int reclen;
89: {
90:     STR temp;
91:     STRPTR p, q, reced;
92:     boolean atf;
93:
94:     atf = !strcmp( index, word );
95:     reced = rec + reclen;
96:     rec += 2 + rec[2] + 1;
97:     for ( p = rec; rec < reced; rec = p ) {
98:         if ( *++p >= 0x20 ) {
99:             for ( q = temp;
100:                 p < reced && *p >= 0x20; *q++ = *p++ );
101:             if ( *rec != classcode )
102:                 continue;
103:             *q = '¥0';
104:             if ( !strcmp( temp, word )
105:                 || atf && !strcmp( temp, "@" ) )
106:                 return ( rec );
107:         } else {
108:             for ( ; p < reced && *++p >= 0x20; );
109:         }
110:     }
111:     return ( NULL );
112: }
113:
114: static int delword1( rec, dword, len, ilen )
115: STRPTR rec, dword;
116: int len, ilen;
117: {
118:     STRPTR p;
119:     int i, j;
120:
121:     for ( i = 1, p = dword + 1;
122:         p < rec + len && *p++ >= 0x20; i++ );
123:     if ( len - 2 - ilen - i - 1 == 0 ) {
124:         i = len;
125:         dword = rec;
126:         len = 0;
127:         *rec = len;
128:     } else {
129:         len -= i;
130:         rec[0] = len;
131:         rec[1] = ( len >> 8 );
132:     }
133:     j = ( pagebuff + PAGELEN ) - ( dword + i );
134:     memcpy( dword, dword + i, j );
135:     memset( dword + j, '¥0', i );
136:
137:     return ( len );
138: }
139:
140: static STAT delword0( index0, word, classcode )
141: STRPTR index0, word;
142: unsigned int classcode;
143: {
144:     STR index;
145:     STAT retcode = ERR;
146:     STRPTR rec, dword;
147:     int len, ilen;
148:     void seekpage( FILE *, int );
149:
150:     SJIStoASK( index, index0 );
151:
152:     if ( ( rec = searchrec( hp, pagebuff, index ) ) == NULL )
153:         return ( NOTFOUND );
154:     len = rec[0] | ( ( ( int ) rec[1] ) << 8 );
155:     ilen = ( int ) rec[2];
156:     if ( ( dword = searchword( rec, index0, word,
157:                 classcode, len ) ) == NULL )
158:         return ( NOTFOUND );
159:     if ( ( len = delword1( rec, dword, len, ilen ) ) != 0
160:         && ( dword = searchword( rec, index0, word,
161:                 classcode, len ) ) != NULL )
162:         delword1( rec, dword, len, ilen );
163:
164:     seekpage( dicfp, curpageno );
165:     if ( fwrite( pagebuff, 1, PAGELEN, dicfp ) != PAGELEN )
166:         diskfull( dicfp );
167:
168:     return ( DONE );
169: }
170:
171: static STAT delword( index, word, class )
172: STRPTR index, word, class;
173: {
174:     unsigned int classcode;
175:     STAT retcode;
176:     int getclasscode( STRPTR );
177:
178:     if ( ( classcode = getclasscode( class ) ) == 0 ) {
179:         verboseprintf( stderr, "¥33[%dm%s¥33[m", ERRCOLOR, li
nbuf );
180:         retcode = ERR;
181:     } else if ( delword0( index, word, classcode ) == DONE )
 {
182:         verboseprintf( stdout, "%s", linbuf );
183:         validline++;
184:         retcode = DONE;
185:     } else {
186:         verboseprintf( stderr, "¥33[%dm%s¥33[m", SKIPCOLOR, l
inbuf );
187:         retcode = NOTFOUND;
188:     }
189:     return ( retcode );
190: }
191:
192: static void windup()
193: {
194:     fputs( "¥33[m", stderr );
195:     if ( invalidline )
196:         fprintf( stderr,
197:         "%d行の無効行がありました¥n", invalidline );
198:     if ( validline )
199:         fprintf( stderr,
200:         "%d個の単語を削除しました¥n", validline );
201:     else
202:         fprintf( stderr, "削除は行われませんでした¥n" );
203: }
204:
205: void main( argc, argv )
206: int argc;
207: STRPTR *argv;
208: {
209:     STR index, word, class;
210:     boolean dflt = TRUE;
211:
212:     int KEYSNS( void );
213:     HEADBUFF *readheader( FILE * );
214:     int getmaxpageno( HEADBUFF * );
215:
216:     for ( ; --argc; ) {
217:         if ( strchr( SWITCH_DELIMITER, **++argv ) != NULL ) {
218:             unsigned int sc;
219:             sc = *++*argv;
220:             switch ( tolower( sc ) ) {
221:                 case '¥0':
222:                     setptr( &wordfile, stdin ); /*dummy*/
223:                     break;
224:                 case 'e':
225:                     setflag( &report );
226:                     break;
227:                 case 'o':
228:                     setptr( &dicfile, ++*argv );
229:                     break;
230:                 case 'v':
231:                     setflag( &verbose );
232:                     break;
233:                 default:
234:                     usage();
235:                     break;
236:             }
237:         } else if ( wordfile == NULL ) {
238:             wordfile = *argv;
239:             dflt = FALSE;
240:         } else if ( dicfile == NULL ) {
241:             dicfile = *argv;
242:         } else {
243:             usage();
244:         }
245:     }
246:     if ( dicfile == NULL )
247:         usage();
248:     else
249:         dicfile = chkext( dicfile, ".DIC" );
250:
251:     if ( ( dicfp = fchkopen( dicfile, "rb" ) ) == NULL )
252:         fatal_error(
253:         "辞書ファイル[%s]がみつかりません", dicfile );
254:
255:     if ( ( dicfp = freopen( dicfile, "r+b", dicfp ) ) == NULL
 )
256:         fatal_error(
```

▶表紙の絵を写真集にでもしたら売れるのではないですか？　私はなんとなく神秘的なところが好きです。
飛石和大(19)東京都

```
257:         "辞書ファイル[%s]は書き込み禁止になっています", dicfi
    le );
258:     hp = readheader( dicfp );
259:     maxpageno = getmaxpageno( hp );
260: 
261:     resetstdin();
262:     if ( dflt ) {
263:         sourfp = stdin;
264:         wordfile = "[標準入力]";
265:     } else if ( ( sourfp = fchkopen( wordfile, "r" ) ) == NUL
    L ) {
266:         fatal_error( ROPENERRMES, wordfile );
267:     }
268: 
269:     if ( report && ( errfp = fopen( errorfile, "w" ) ) == NUL
    L )
270:         fatal_error( EOPENERRMES, errorfile );
271: 
272:     breakset( windup );
273: 
274:     fprintf( stderr,
275:         "削除単語ファイル     :%s¥n削除対象辞書ファイル:%s¥n
    "
276:         wordfile, dicfile );
277: 
278:     for ( *index = '¥0', linctr = 1;
279:         fgets( linbuf, NCHAR, sourfp  ) != NULL; linctr++ ) {
280: 
281:         KEYSNS();
282:         if ( !parseline( linbuf, index, word, class )
283:             || delword( index, word, class ) == ERR ) {
284:             invalidline++;
285:             if ( report && fputs( linbuf, errfp ) == EOF )
286:                 diskfull( errfp );
287:             *index = '¥0';
288:         }
289:     }
290: 
291:     fclose( sourfp );
292:     fclose( dicfp );
293:     if ( report )
294:         fclose( errfp );
295: 
296:     windup();
297:     exit( EXIT_SUCCESS );
298: }
```

リスト5 SPLIT.C

```
  1: /*
  2:         SPLIT.X
  3:         テキストを複数ファイルに分割するフィルタ
  4: 
  5:         split.c misc.c
  6:         vfprintf.s
  7:         mydef.h misc.h myerror.h
  8:         doslib.a
  9: */
 10: 
 11: #include    "mydef.h"
 12: #include    <stdio.h>
 13: #include    <stdlib.h>
 14: #include    <string.h>
 15: #include    <limits.h>
 16: #include    <doslib.h>
 17: #include    "misc.h"
 18: #include    "myerror.h"
 19: 
 20: #define     NLINE       1000
 21: #define     TEMPNAME    "SPLTEMP."
 22: 
 23: PACKEDSTR progname = "SPLIT";
 24: PACKEDSTR usagemes =
 25: "機  能:テキストを複数のファイルに分割します¥n¥
 26: 使用法:%s [スイッチ] [入力ファイル] [[/O]出力先ディレクトリ]¥n¥
 27: ¥t/Bn¥tそれぞれがnバイト以下になるように分割する¥n¥
 28: ¥t¥t(ただし,各ファイルに少なくとも1行は出力する)¥n¥
 29: ¥t/Ln¥tn行ごとに分割する¥n¥
 30: ¥t/Wn¥t1行の最大文字数をn文字にする¥n¥n¥
 31: ¥t/Bも/Lも省略された場合は/L1000とみなす¥n";
 32: 
 33: FILE *nextfile( destfile, no )
 34: STRPTR destfile;
 35: int no;
 36: {
 37:     STR temp;
 38:     FILE *fp;
 39: 
 40:     if ( no > 0xfff )
 41:         fatal_error( "最大分割数を越えました" );
 42:     sprintf( temp, "%s%.3X", destfile, no );
 43:     if ( ( fp = fopen( temp, "w" ) ) == NULL )
 44:         fatal_error( WOPENERRMES, temp );
 45:     return ( fp );
 46: }
 47: 
 48: void main( argc, argv )
 49: int argc;
 50: STRPTR *argv;
 51: {
 52:     STRPTR linbuf;
 53:     boolean blflg = FALSE, dflt = TRUE;
 54:     long divbyt = 0, bytctr, linlen;
 55:     long divlin = 0, linctr;
 56:     int no = 0, lwidth = 0;
 57:     FILE *fp1, *fp2;
 58:     STRPTR sourfile = NULL, destfile = NULL;
 59: 
 60:     for ( ; --argc; ) {
 61:         if ( strchr( SWITCH_DELIMITER, **++argv ) != NULL ) {
 62:             unsigned int sc;
 63:             sc = *++*argv;
 64:             switch ( tolower( sc ) ) {
 65:                 case '¥0':
 66:                     setptr( &sourfile, stdin ); /*dummy*/
 67:                     break;
 68:                 case 'b':
 69:                     setflag( &blflg );
 70:                     divlin = LONG_MAX;
 71:                     setvalue( &divbyt, *argv, 1, LONG_MAX - 1 );
 72:                     break;
 73:                 case 'l':
 74:                     setflag( &blflg );
 75:                     divbyt = LONG_MAX;
 76:                     setvalue( &divlin, ++*argv, 1, LONG_MAX - 1
 );
 77:                     break;
 78: #if 0
 79:                 case 'n':
 80:                     setvalue( &no, ++*argv, 0, 0xfff );
 81:                     break;
 82: #endif
 83:                 case 'o':
 84:                 }   setptr( &destfile, ++*argv );
 85:                     break;
 86:                 case 'w':
 87:                     setvalue( &lwidth, ++*argv, NCHAR, INT_MAX )
 ;
 88:                     break;
 89:                 default:
 90:                     usage();
 91:                     break;
 92:             }
 93:         } else if ( sourfile == NULL ) {
 94:             sourfile = *argv;
 95:             dflt = FALSE;
 96:         } else if ( destfile == NULL ) {
 97:             destfile = *argv;
 98:         } else {
 99:             usage();
100:         }
101:     }
102: 
103:     resetstdin();
104:     if ( dflt ) {
105:         fp1 = stdin;
106:         destfile = TEMPNAME;
107:     } else {
108:         struct  NAMECKBUF nambuf;
109:         if ( NAMECK( sourfile, &nambuf ) ) {
110:             fatal_error( ILLNAMMES, sourfile );
111:         } else if ( ( fp1 = fopen( sourfile, "r" ) ) == NULL ) {
112:             fatal_error( ROPENERRMES, sourfile );
113:         } else {
114:             if ( destfile == NULL )
115:                 sprintf( linbuf, "%s.", nambuf.name );
116:             else if ( strchr( PATH_DELIMITER, *( strchr( destfil
    e, '¥0' )- 1 ) ) == NULL )
117:                 sprintf( linbuf, "%s¥¥%s.", destfile, nambuf.nam
    e );
118:             else
119:                 sprintf( linbuf, "%s%s.", destfile, nambuf.name
    );
120:             destfile = dupstr( linbuf );
121:         }
122:     }
123: 
124:     if ( divbyt + divlin == 0 ) {
125:         divbyt = LONG_MAX;
126:         divlin = NLINE;
127:     }
128:     if ( lwidth == 0 )
129:         lwidth = NCHAR;
130:     linbuf = memalloc( lwidth );
131: 
132:     breakset( NULL );
133: 
134:     fp2 = nextfile( destfile, no++ );
135: 
136:     for ( bytctr = linctr = 0; fgets( linbuf, lwidth, fp1 ) !=
    NULL;
137:                                       bytctr += linlen, linctr++ )
    {
138:         linlen = strlen( linbuf ) + 1;
139:         if ( bytctr != 0 && bytctr + linlen > divbyt
140:                                       || ( linctr >= divlin ) ) {
141:             fclose( fp2 );
142:             bytctr = linctr = 0;
143:             fp2 = nextfile( destfile, no++ );
144:         }
145:         if ( fputs( linbuf, fp2 ) == EOF )
146:             diskfull( fp2 );
147:     }
148:     fcloseall();
149: 
150:     exit( EXIT_SUCCESS );
151: }
```

▶がんばれ！　アルバイター。夜勤がなんだ。X68000君は目の前だ～っ！
坊農誠(19)福井県

リスト6 PARSELINE.C

```
 1: #include    "mydef.h"
 2: #include    <stdio.h>
 3: #include    <ctype.h>
 4: #include    <string.h>
 5: #include    <jstring.h>
 6:
 7: boolean parseline( linbuf, index, word, class )
 8: STRPTR linbuf, index, word, class;
 9: {
10:     int len;
11:     STRPTR p;
12:
13:     *word = *class = '¥0';
14:     if ( isspace( *linbuf ) ) {
15:         sscanf( linbuf, "%s%s", word, class );
16:     } else {
17:         sscanf( linbuf, "%s%s%s", index, word, class );
18:         if ( *word == '¥0'
19:             && ( p = jstrrchr( index, '：' ) ) ) {
20:             strcpy( word, p );
21:             *p = '¥0';
22:         }
23:         if ( ( len = strlen( index ) ) > 2
24:             && !jstrcmp( p = index + len - 2, "：" ) )
25:             *p = '¥0';
26:         strzenkata( index, index );
27:     }
28:     if ( *class == '¥0'
29:         && ( p = jstrrchr( word, '／' ) ) ) {
30:         strcpy( class, p + 2 );
31:         *p = '¥0';
32:     }
33:     if ( !strncmp( class, "／", 2 ) )
34:         strcpy( class, class + 2 );
35:
36:     return ( *index != '¥0'
37:             && *word != '¥0' && *class != '¥0' );
38: }
```

リスト7 PAGENO.C

```
 1: #include    "mydef.h"
 2: #include    <stdio.h>
 3: #include    <io.h>
 4: #include    <string.h>
 5: #include    "misc.h"
 6: #include    "myerror.h"
 7: #include    "dictools.h"
 8:
 9: static PACKEDSTR ILLFORMMES =
10: "指定のファイルはASK68Kの辞書ファイル¥
11: ではありません（たぶん）";
12:
13: HEADBUFF *readheader( fp )
14: FILE *fp;
15: {
16:     HEADBUFF *hp;
17:     int len;
18:
19:     hp = memalloc( sizeof( HEADBUFF ) );
20:     len = filelength( fileno( fp ) );
21:     if ( len < PAGELEN + sizeof( HEADBUFF )
22:         || len % PAGELEN != 0 )
23:         fatal_error( ILLFORMMES );
24:     if ( fread( hp, 1, sizeof( HEADBUFF ), fp )
25:                         != sizeof( HEADBUFF ) )
26:         fatal_error( RERRORMES );
27:     return ( hp );
28: }
29:
30: int getmaxpageno( hp )
31: HEADBUFF *hp;
32: {
33:     INDEXTYPE *p;
34:     int maxpageno;
35:
36:     for ( maxpageno = 0, p = hp->index;
37:         p < hp->index + MAXINDEX && **p != '¥0'; p++, maxpage
no++ );
38:     return ( maxpageno );
39: }
40:
41: int getpageno( hp, str )
42: HEADBUFF *hp;
43: STRPTR str;
44: {
45:     int pageno;
46:     INDEXTYPE temp, *p;
47:
48:     memset( temp, '¥0', sizeof( INDEXTYPE ) );
49:     strncpy( temp, str, sizeof( INDEXTYPE ) );
50:
51:     for ( pageno = 0, p = hp->index;
52:         p < hp->index + MAXINDEX && **p != '¥0'; p++, pageno+
+ ) {
53:         if ( memcmp( p, temp, sizeof( INDEXTYPE ) ) >= 0 )
54:             break;
55:     }
56:     return ( ( pageno != 0 ) ? pageno - 1 : 0 );
57: }
58:
59: void seekpage( fp, pageno )
60: FILE *fp;
61: int pageno;
62: {
63:     if ( fseek( fp, pageno * PAGELEN + sizeof( HEADBUFF ), SE
EK_SET ) )
64:         fatal_error( ILLFORMMES );
65: }
```

リスト8 SELCLASS.C

```
 1: #include    "mydef.h"
 2: #include    <doslib.h>
 3: #include    <iocslib.h>
 4: #include    "dictools.h"
 5:
 6: #define     NORMCOLOR   3
 7: #define     REVCOLOR    13
 8:
 9: #define     ESC     ( 0x01 )
10: #define     BREAK   ( 0x61 )
11: #define     CR      ( 0x1D )
12: #define     ENTER   ( 0x4e )
13: #define     CUP     ( 0x3c )
14: #define     CDOWN   ( 0x3e )
15: #define     CRIGHT  ( 0x3d )
16: #define     CLEFT   ( 0x3b )
17: #define     SPACE   ( 0x35 )
18:
19: #define     MENULEN     18
20:
21: unsigned int selclass( flags )
22: unsigned int flags;
23: {
24:     unsigned int b;
25:     int m=0, savpos;
26:
27:     static STRPTR class[] = {
28: /*          123456789012345678      */
29:             "　　　カ五動詞　　",
30:             "　　　ガ五動詞　　",
31:             "　　　サ五動詞　　",
32:             "　　　タ五動詞　　",
33:             "　　　ナ五動詞　　",
34:             "　　　バ五動詞　　",
35:             "　　　マ五動詞　　",
36:             "　　　ラ五動詞　　",
37:             "　　　ワ五動詞　　",
38:             "　　　サ変動詞　　",
39:             "　　　カ変動詞　　",
40:             "　上・下一段動詞　",
41:             "　　　形　容　詞　",
42:             "　　　形容動詞　　",
43:             "　形容動詞複合名詞",
44:             "　　サ変複合名詞　",
45:             "　　　名　　　詞　",
46:             "　　　単　漢　字　",
47:             "　　　人名（姓）　",
48:             "　　　人名（名）　",
49:             "　　　地　　　名　",
50:             "　　　団　体　名　",
51:             "　　　物の名称　　",
52:             "　　　数　　　詞　",
53:             "　　　数　　　字　",
54:             "　　　接　尾　語　",
55:             "　　　感　動　詞　",
56:             "　　　接　続　詞　",
57:             "　　　副　　　詞　",
58:             "　　　連　体　詞　",
59:     };
60:
61:     if ( C_WIDTH( -1 ) > 1 )
62:         C_WIDTH( 0 );
63:     C_COLOR( NORMCOLOR );
64:     for ( m = 0; m < 30; m++ ) {
65:         if ( flags & ( 1 << ( m + 1 ) ) )
66:             C_COLOR( REVCOLOR );
67:         B_PRINT( class[ m ] );
68:         C_COLOR( NORMCOLOR );
69:         if ( m % 4 == 3 )
70:             B_PRINT( "¥r¥n" );
71:     }
72:     B_PRINT( "¥r¥n" );
73:
74:     C_CURON();
75:     savpos = C_LOCATE( -1, -1 );
76:     C_UP( 8 );
77:
78:     for ( ; B_KEYSNS(); B_KEYINP() );
79:
80:     for ( m = 0;; ) {
81:         switch ( ( B_KEYINP() >> 8 ) & 0xff ) {
82:             case ESC:
83:             case BREAK:
84:                 C_LOCATE( savpos >> 16, savpos );
85:                 return ( 0 );
86:                 break;
87:             case CR:
88:             case ENTER:
89:                 C_LOCATE( savpos >> 16, savpos );
90:                 return ( flags );
```

▶そういえばOh!MZの最初の頃は結構過激なイラストでしたね。書店で見た覚えがあります。
和田安生(21)福岡県

```
 91:                 break;
 92:             case CUP:
 93:                 if ( m >= 4 ) {
 94:                     m -= 4;
 95:                     C_UP_S();
 96:                 }
 97:                 break;
 98:             case CDOWN:
 99:                 if ( m <= 25 ) {
100:                     m += 4;
101:                     C_DOWN_S();
102:                 }
103:                 break;
104:             case CRIGHT:
105:                 if ( m <= 28 && m % 4 < 3 ) {
106:                     m++;
107:                     C_RIGHT( MENULEN );
108:                 }
109:                 break;
110:             case CLEFT:
111:                 if ( m % 4 ) {
112:                     m--;
113:                     C_LEFT( MENULEN );
114:                 }
115:                 break;
116:             case SPACE:
117:                 b = 1 << ( m + 1 );
118:                 if ( flags & b ) {
119:                     flags &= ~b;
120:                 } else {
121:                     flags |= b;
122:                     C_COLOR( REVCOLOR );
123:                 }
124:                 B_PRINT( class[ m ] );
125:                 C_LEFT( MENULEN );
126:                 C_COLOR( NORMCOLOR );
127:                 break;
128:         }
129:     }
130: }
```

リスト9 WCLASS.C

```
 1: #include    "mydef.h"
 2: #include    <string.h>
 3:
 4: STRPTR classtbl[] = {
 5:     "",                     "カ五動詞",
 6:     "ガ五動詞",             "サ五動詞",
 7:     "タ五動詞",             "ナ五動詞",
 8:     "バ五動詞",             "マ五動詞",
 9:     "ラ五動詞",             "ワ五動詞",
10:     "サ変動詞",             "カ変動詞",
11:     "上・下一段動詞",       "形容詞",
12:     "形容動詞",             "形容動詞複合名詞",
13:     "サ変複合名詞",         "名詞",
14:     "単漢字",               "人名(姓)",
15:     "人名(名)",             "地名",
16:     "団体名",               "物の名称",
17:     "数詞",                 "数字",
18:     "接尾語",               "感動詞",
19:     "接続詞",               "副詞",
20:     "連体詞" ,
21: };
22:
23: int getclasscode( s )
24: STRPTR s;
25: {
26:     int i;
27:
28:     for ( i = 1; i < 31; i++ ) {
29:         if ( !strcmp( s, classtbl[ i ] ) )
30:             return ( i );
31:     }
32:     return ( 0 );
33: }
```

リスト10 STRFUNC.C

```
  1: #include    "mydef.h"
  2: #include    <string.h>
  3: #include    <ctype.h>
  4: #include    <jfctype.h>
  5: #include    <iocslib.h>
  6:
  7: void settab( s, n )
  8: STRPTR s;
  9: int n;
 10: {
 11:     n -= strlen( s );
 12:     s = strchr( s, '¥0' );
 13:     do {
 14:         *s++ = '¥t';
 15:         n -= 8;
 16:     } while ( n > 0 );
 17:     *s = '¥0';
 18: }
 19:
 20: void strtoupper( s )
 21: STRPTR s;
 22: {
 23:     WCHAR c;
 24:
 25:     for ( ; c = *s; ) {
 26:         if ( iskanji( c ) ) {
 27:             if ( *++s == '¥0' )
 28:                 break;
 29:             s++;
 30:         } else {
 31:             *s++ = toupper( c );
 32:         }
 33:     }
 34: }
 35:
 36: STRPTR strzentohan( dest, sour )
 37: STRPTR dest, sour;
 38: {
 39:     STRPTR p = dest;
 40:     WCHAR c, ch;
 41:
 42:     for ( ; c = *sour++; ) {
 43:         if ( iskanji( c ) ) {
 44:             if ( !iskanji2( *sour ) )
 45:                 break;
 46:             c = zentohan( c << 8 | *sour++ );
 47:         }
 48:         switch ( ch = c >> 8 ) {
 49:             case 0:
 50:                 *p++ = c;
 51:                 break;
 52:             case 0xde:  /*  「ﾞ」 */
 53:             case 0xdf:  /*  「ﾟ」 */
 54:                 *p++ = c;
 55:                 *p++ = ch;
 56:                 break;
 57:             default:
 58:                 *p++ = ch;
 59:                 *p++ = c;
 60:                 break;
 61:         }
 62:     }
 63:     *p = '¥0';
 64:
 65:     return ( dest );
 66: }
 67:
 68: #define     DHIRAKATA   ( 0x100 )
 69:
 70: STRPTR strhantozen( dest, sour, hiragana )
 71: STRPTR dest, sour;
 72: boolean hiragana;
 73: {
 74:     STRPTR p = dest, temp = NULL;
 75:     WCHAR c;
 76:     STRPTR dupstr( STRPTR );
 77:
 78:     if ( sour == dest )
 79:         temp = sour = dupstr( sour );
 80:
 81:     for ( ; c = *sour++; ) {
 82:         if ( iskanji( c ) ) {
 83:             if ( !iskanji2( *sour ) )
 84:                 break;
 85:             c = ( c << 8 ) + *sour++;
 86:             if ( !hiragana && ( jishira( c ) ) )
 87:                 c = JISSFT( SFTJIS( c ) + DHIRAKATA );
 88:             else if ( hiragana && ( jiskata( c ) ) )
 89:                 c = JISSFT( SFTJIS( c ) - DHIRAKATA );
 90:             *( ( WCHAR * ) p )++ = c;
 91:         } else if ( c == 0xde ) {
 92:             p += DAKJOB( p ) * sizeof( WCHAR );
 93:         } else if ( c == 0xdf ) {
 94:             p += HANJOB( p ) * sizeof( WCHAR );
 95:         } else {
 96:             *( ( WCHAR * ) p )++ = hantozen( c | hiragana <<
   15 );
 97:         }
 98:     }
 99:     *p = '¥0';
100:
101:     if ( temp != NULL )
102:         free( temp );
103:
104:     return ( dest );
105: }
106:
107: STRPTR strzenkata( dest, sour )
108: STRPTR dest, sour;
109: {
110:     return( strhantozen( dest, sour, FALSE ) );
111: }
112:
113: STRPTR strzenhira( dest, sour )
114: STRPTR dest, sour;
115: {
116:     return( strhantozen( dest, sour, TRUE ) );
117: }
118:
119: STRPTR SJIStoASK( dest, sour )
120: STRPTR dest, sour;
121: {
122:     STRPTR p = dest;
123:     WCHAR c;
124:
125:     for ( ; *sour; ) {
126:         switch ( c = *( ( WCHAR * ) sour ) ++ ) {
127:             case 0x8142:
```

▶シャープの広告の右上隅のツタンカーメンの横の女の人の顔が幽霊みたいで怖い。
颯々野学(24)大阪府

```
128:                c = 0x98;   break;
129:            case 0x8175:
130:                c = 0x99;   break;
131:            case 0x8176:
132:                c = 0x9a;   break;
133:            case 0x8141:
134:                c = 0x9b;   break;
135:            case 0x8145:
136:                c = 0x9c;   break;
137:            case 0x815b:
138:                c = 0x9d;   break;
139:            case 0x814a:
140:                c = 0x9e;   break;
141:            case 0x814b:
142:                c = 0x9f;   break;
143:            default:
144:                if ( c < 0x829f )
145:                    c = zentohan( c );
146:                else if ( c < 0x8340 )
147:                    c = SFTJIS( c ) - 0x2421 + 0xa1;
148:                else if ( c < 0x8397 )
149:                    c = SFTJIS( c ) - 0x2521 + 0xa1;
150:                else
151:                    c = '¥0';
152:                break;
153:        }
154:        *p++ = c;
155:    }
156:    *p = '¥0';
157:
158:    return ( dest );
159: }
160:
161: STRPTR ASKtoSJIS( dest, sour )
162: STRPTR dest, sour;
163: {
164:    static WCHAR spctbl[] = {
165:        0x8142, 0x8175, 0x8176, 0x8141,
166:        0x8145, 0x815b, 0x814a, 0x814b,
167:    };
168:    STRPTR p = dest, temp = NULL;
169:    WCHAR c;
170:    STRPTR dupstr( STRPTR );
171:
172:    if ( sour == dest )
173:        temp = sour = dupstr( sour );
174:
175:    for ( ; c = *sour++; ) {
176:        if ( c < 0x80 )
177:            c = hantozen( c );
178:        else if ( ( 0xa1 <= c ) && ( c <= 0xdf ) )
179:            c += 0x8340 - 0xa1;
180:        else if ( ( 0xe0 <= c ) && ( c <= 0xf6 ) )
181:            c += 0x8380 - 0xe0;
182:        else if ( ( 0x98 <= c ) && ( c <= 0x9f ) )
183:            c = spctbl[ c - 0x98 ];
184:        else
185:            c = 0x81a6;
186:        *( ( WCHAR *) p )++ = c;
187:    }
188:    *p = '¥0';
189:
190:    if ( temp != NULL )
191:        free( temp );
192:
193:    return ( dest );
194: }
195:
196: void setindex( dest, sour, hankaku, hiragana )
197: STRPTR dest, sour;
198: boolean hankaku, hiragana;
199: {
200:    if ( hankaku )
201:        strzentohan( dest, sour );
202:    else if ( hiragana )
203:        strzenhira( dest, sour );
204:    else
205:        strcpy( dest, sour );
206:    settab( dest, 8 * 3 );
207: }
```

リスト11 MISC.C

```
  1: #include    "mydef.h"
  2: #include    <stdio.h>
  3: #include    <stdlib.h>
  4: #include    <io.h>
  5: #include    <string.h>
  6: #include    <ctype.h>
  7: #include    <jfctype.h>
  8: #include    <doslib.h>
  9: #include    <signal.h>
 10: #include    "myerror.h"
 11:
 12: typedef void *va_list;
 13: #define va_start( ap, parmN )¥
 14:     ( ap = ( ( ( va_list * ) ( & ( parmN ) ) ) + 1 ) )
 15: #define va_end( ap )
 16:
 17: extern PACKEDSTR progname;
 18: extern PACKEDSTR usagemes;
 19:
 20: void usage()
 21: {
 22:     fprintf( stderr, usagemes, progname );
 23:     exit( EXIT_FAILURE );
 24: }
 25:
 26: #ifdef __STDC__
 27: void fatal_error( STRPTR format, ... )
 28: #else
 29: void fatal_error( format, )
 30: STRPTR format;
 31: #endif
 32: {
 33:     boolean uf;
 34:     va_list argptr;
 35:     int vfprintf( FILE *, STRPTR, va_list );
 36:
 37:     va_start( argptr, format );
 38:     fprintf( stderr, "%s : ", progname );
 39:     if ( uf = ( *format < '¥x20' ) )
 40:         format++;
 41:     vfprintf( stderr, format, argptr );
 42:     fputc( '¥n', stderr );
 43:     va_end( argptr );
 44:     if ( uf )
 45:         usage();
 46:     exit( EXIT_FAILURE );
 47: }
 48:
 49: void diskfull( fp )
 50: FILE *fp;
 51: {
 52:     int devinfo;
 53:
 54:     devinfo = IOCTRLGT( fileno( fp ) );
 55:     if ( devinfo & ( 1 << 7 ) )
 56:         fatal_error( "出力できません" );
 57:     else
 58:         fatal_error( "ディスク[ドライブ%c:]がいっぱいです",
 59:         ( devinfo & 0x1f ) + 'A' );
 60: }
 61:
 62: static void (*cleanupfunc)( void ) = NULL;
 63:
 64: void abt()
 65: {
 66:     if ( cleanupfunc != NULL )
 67:         (*cleanupfunc)();
 68:     fatal_error( "中断しました¥n" );
 69: }
 70:
 71: void breakset( func )
 72: void (*func)();
 73: {
 74:     cleanupfunc = func;
 75:     signal( SIGINT, abt );
 76: }
 77:
 78: void *memalloc( size )
 79: unsigned int size;
 80: {
 81:     void *p;
 82:
 83:     if ( size == 0 )
 84:         size = 2;
 85:     else if ( size & 1 )
 86:         size++;
 87:     for ( ; ( p = malloc( size ) ) == NULL; ) {
 88:         if ( sbrk( size + 16 ) == ( void * ) -1 )
 89:             fatal_error( NOMEMMES );
 90:     }
 91:     return ( p );
 92: }
 93:
 94: void memtest( size )
 95: unsigned int size;
 96: {
 97:     free( memalloc( size ) );
 98: }
 99:
100: STRPTR dupstr( str )
101: STRPTR str;
102: {
103:     STRPTR p;
104:     int strlen( STRPTR );
105:     p = memalloc( strlen( str ) + 1 );
106:     return ( strcpy( p, str ) );
107: }
108:
109: STRPTR chkext( fname, ext )
110: STRPTR fname, ext;
111: {
112:     STR name;
113:
114:     strcpy( name, fname );
```

▶ドクター・パソコンはいま，どこで，何をしておられるのだろう。　佐伯章(17)東京都

```
115:     if ( !strchr( fname, '.' ) )
116:         strcat( name, ext );
117:     return ( dupstr( name ) );
118: }
119:
120: FILE *fchkopen( fname, mode )
121: STRPTR fname, mode;
122: {
123:     struct  NAMECKBUF nambuf;
124:
125:     if ( NAMECK( fname, &nambuf ) != 0 )
126:         fatal_error( ILLNAMMES, fname );
127:     return ( fopen( fname, mode ) );
128: }
129:
130: void setflag( swp )
131: boolean *swp;
132: {
133:     if ( (*swp)++ )
134:         usage();
135: }
136:
137: void setvalue( vp, str, min, max )
138: int *vp;
139: STRPTR str;
140: int min, max;
141: {
142:     if ( *vp != 0 || ( *vp = atoi( str ) ) < min || *vp > max
 )
143:         usage();
144: }
145:
146: void setptr( dest, sour )
147: void **dest, *sour;
148: {
149:     if ( *dest != NULL )
150:         usage();
151:     *dest = sour;
152: }
153:
154: void resetstdin()
155: {
156:     int fno1, fno2;
157:
158:     fno2 = dup( fno1 = fileno( stdin ) );
159:     close( fno1 );
160:     stdin->_file = fno2;     /***/
161: }
162:
163: FILE *setstdout( name )
164: STRPTR name;
165: {
166:     FILE *fp;
167:
168:     if ( name != NULL ) {
169:         if ( ( fp = fchkopen( name, "w" ) ) == NULL )
170:             fatal_error( WOPENERRMES, name );
171:     } else {
172:         fp = stdout;
173:     }
174:
175:     return ( fp );
176: }
```

リスト12 VFPRINT.S

```
 1:         .xdef   _vfprintf
 2:         .xdef   _fprintf
 3:         .xdef   _printf
 4:
 5:         .xref   _fputc
 6:         .xref   __fmtout
 7:         .xref   __iob
 8: *
 9: EOF     equ     -1
10: *
11:         .text
12: *
13: _vfprintf:
14:         move.l  (sp)+,ret+2
15:         pea.l   _fputc
16:         jsr     __fmtout
17:         addq.l  #4,sp
18:         movea.l (sp),a0          *FILE *
19:         btst.b  #6,15(a0)        *_IOERR
20:         beq     ret
21:
22:         moveq.l #EOF,d0
23:
24: ret:    jmp     0
25:
26: *
27: _fprintf:
28:         pea.l   12(sp)
29:         move.l  8+4(sp),-(sp)
30:         move.l  4+8(sp),-(sp)
31:         bsr     _vfprintf
32:         lea.l   12(sp),sp
33:         rts
34:
35: *
36: _printf:
37:         pea.l   8(sp)
38:         move.l  4+4(sp),-(sp)
39:         pea.l   __iob+22
40:         bsr     _vfprintf
41:         lea.l   12(sp),sp
42:         rts
43:
44:         .end
```

リスト13 MYDEF.H

```
 1: #define     SWITCH_DELIMITER    "/-"
 2: #define     PATH_DELIMITER      "¥¥/"
 3:
 4: #define     NULL        ( ( void * ) 0 )
 5:
 6: #define     EXIT_SUCCESS 0
 7: #define     EXIT_FAILURE 1
 8:
 9: #define     SEEK_SET    0
10: #define     SEEK_CUR    1
11: #define     SEEK_END    2
12:
13: #define     NCHAR       256
14: typedef     unsigned char   *STRPTR,
15:                             PACKEDSTR[],
16:                             STR[NCHAR];
17:
18: typedef     unsigned short  WCHAR;
19:
20: typedef     enum { FALSE, TRUE, } boolean;
21:
22: #define     S       ( STRPTR )
23:
24: #define     MACRO   /*  for stdio.h */
```

リスト14 MISC.H

```
 1: void *memalloc( unsigned int );
 2: void memtest( unsigned int );
 3: STRPTR dupstr( STRPTR );
 4: STRPTR chkext( STRPTR, STRPTR );
 5: FILE *fchkopen( STRPTR, STRPTR );
 6: FILE *setstdout( STRPTR );
 7: void resetstdin( void );
 8: void setflag( boolean * );
 9: void setvalue( int *, STRPTR, int, int );
10: void setptr( void **, void * );
```

リスト15 STRFUNC.H

```
 1: void settab( STRPTR, int );
 2: void strtoupper( STRPTR );
 3: STRPTR strhantozen( STRPTR, const STRPTR, boolean );
 4: STRPTR strzenhira( STRPTR, const STRPTR );
 5: STRPTR strzenkata( STRPTR, const STRPTR );
 6: STRPTR strzentohan( STRPTR, const STRPTR );
 7: STRPTR ASKtoSJIS( STRPTR, const STRPTR );
 8: STRPTR SJIStoASK( STRPTR, const STRPTR );
 9: void setindex( STRPTR, const STRPTR, boolean, boolean );
```

リスト16 MYERROR.H

```
 1: #ifdef  __STDC__
 2:     void fatal_error( const STRPTR, ... );
 3: #else
 4:     void fatal_error( STRPTR, );
 5: #endif
 6: void diskfull( FILE * );
 7: void usage( void );
 8: void breakset( void (*)() );
 9:
10: #define     NOMEMMES    "作業用メモリが足りません"
11: #define     ILLNAMMES   "¥1ファイル名の指定[%s]に誤りがあります"
12: #define     ROPENERRMES "指定のファイル[%s]がみつかりません"
13: #define     WOPENERRMES "ファイル[%s]が作成できません"
14: #define     RERRORMES   "ファイルがうまく読み込めません"
15: #define     TOPENERRMES "作業用ファイルが作成できません"
16: #define     EOPENERRMES "エラーファイル[%s]が作成できません"
```

リスト17 DICTOOLS.H

```
 1: #define     PAGELEN     1024
 2: #define     INDEXLEN    8
 3: #define     MAXINDEX    ( 0x1f00 / INDEXLEN )
 4:
 5: typedef unsigned char   PAGEBUFF[ PAGELEN ],
 6:                         INDEXTYPE[ INDEXLEN ];
 7:
 8: typedef struct {
 9:     INDEXTYPE index[ MAXINDEX ];
10:     unsigned char flag[ 0xff ];
11:     char version;
12: } HEADBUFF;
13:
14: enum dicttypes {
15:     DFLT, ASK, E1, MATU,
16: };
```

PASCALプログラミングへの招待〈4〉

PASCALの制御構造,関数および手続き

データ構造に続き,今回はプログラミングの基本となる制御構造からPASCALの特長を見ていきましょう。おまけとして逆ポーランドから通常の式への変換法のサンプルプログラムも掲載します。

Fujiki Takeshi/Fujii Yoshimi
藤木健士/藤井義巳

連載4回目になりました。今回はアルゴリズムをプログラミング言語で記述するのに必要なifやwhileといった条件による分岐や繰り返しのための制御構造と,関数,手続きの宣言および呼び出し方を説明します。もうすでに皆さんはいくつかプログラムを書いてみられたと思いますので,もう知っておられることも多いと思いますが,整理するという意味も兼ねて詳細を説明することにします。

制御構造について

電子計算機の歴史がまだ浅い頃,プログラマは少しでもスピードを上げるために,また少ないメモリを節約してプログラムを1ステップでも短くするために,もはや自分だけしかわからないような非常に技巧的なプログラムを書いたものです。それはもう凄まじいものがありました。最たるものではプログラムが自分自身のロードされているメモリにアクセスして自分自身を書き換えながら別の処理を行うようなプログラムを見たことがあります。

昔はソフトウェアがハードウェアに強く依存し,小規模なシステムしか存在しなかったのでそれでもよかったのでしょう。しかも処理速度は非常に遅くメモリなどの資源の値段が非常に高かったのでしかたありませんでした。しかし,最近はハードウェアが安くなり,計算機の処理能力が格段によくなってきています。またソフトウェアに対する認識が変化して,処理の方法自体に価値が存在するようになってきました。こうなってくると,誰にでもわかりやすいプログラムを書くことが非常に重要になってきます。最近ではプログラムをわかりやすくするための構造化プログラミングはもはや常識といっていいでしょう。

PASCALの設計者ヴィルトもダイクストラが提唱した構造化プログラミングを強く意識していたようで,PASCALにはそれを可能とする制御構造が含まれています。それでは,今回はこの制御構造に関して説明することにします。

制御構造は選択と繰り返しの2つに分けることができます。PASCALには条件分岐はifとcase,繰り返しにはwhileとforとrepeatが用意されています。あとgotoも使うことができます。それではそれぞれについて説明していきたいと思います。

if

ifはもっとも多く使われるステートメントで,これを使えない手続き型の言語は存在しないといってもよいほどです。読者の皆さんにも馴染みの深いものでしょう。if文のかたちは次のとおりです。

```
if　条件式　then　文
```

または,

```
if　条件式　then　文　else　文
```

条件式は論理型の結果を持つ式でなければなりません。ですからここには結果が真(true)か偽(false)になる式を書いてください。たとえば次のような感じです。

```
if i > 1 then
      str := 'CLARIS';
```

あるいはSWという変数が論理型(忘れた方は先月号を見てください)で宣言されているとすると,次のような書き方をすることもできます。

```
var sw  boolean;
    sw := i > 1;
    if sw then
          str := 'ADDIE';
```

複数の文を実行したいときはbeginとendで括って次のように書いてください。

```
if j = 0 then
  begin
     str1 := 'SILKIA';
     str2 := 'CHACKN'
  end;
```

elseを指定して条件式が偽だったときに行う処理を次のように指定することもできます。

```
if rc <> 0 then
              name := 'Ohno'
         else
              name := 'Takeuchi';
```

このとき注意しなければならないのは,elseの前に';'(セミコロン)をつけてはならないということです。PASCALではelseの前にはセミコロンがくることはありません。これは非常に重要なので覚えておいてください。

case

caseは多岐選択に用いられる制御構造で次のようなかたちをしています。

```
case     式     of
定数,定数,……,定数：   文;
定数,定数,……,定数：   文;
          ⋮
定数,定数,……,定数：   文
end
```

ただし式は順序型（整数型，論理型，文字型および列挙型，先月号参照)でなければなりません。if文では真か偽かで2つにひとつの選択しかできないので，値によって3つ以上の異なる処理を行いたいとき，記述が繁雑になることがあります。たとえば次のような場合にはどうでしょうか。

```
if num = 1 then
    day := 'Sun'
else if num = 2 then
    day := 'Mon'
          ⋮
else if num = 6 then
    day := 'Fri'
else
    day := 'Sat,;
```

これと同様の処理をcase文を用いて書くと次のようになります。

```
case num of
    1:     day := 'Sun';
    2:     day := 'Mon';
              ⋮
    5:     day := 'Fri';
    6:     day := 'Sat'
end;
```

ずいぶんすっきりしたでしょう！　定数のあとに書かれる文はひとつのステートメントかbeginとendで囲まれた複数のステートメントで構成されます。それで，case文でひとつの定数に対して複数の処理を行いたいときは':'（コロン）のあとbegin～endで囲まれた複数のステートメントを書いてください。

while

次は条件が満たされているときに繰り返す命令です。次のようなかたちをしています。

```
while   式   do   文
```

式は結果が論理型の値を持たなければなりません。式がtrueである限り指定された文の実行を繰り返します。次の例はiの階乗を求めるものです。

```
var    i, fact : integer
read(i);
fact := 1;
while i > 1 do
    begin
        fact := fact * i;
        i := i-1
    end;
write('fact=', fact);
```

このプログラムは最初にi＞1を評価してそれが真であったらbegin～endで囲まれた部分の処理を行います。それでiが1より小さな値が読み込まれたときには，1を表示することになります。

repeat

repeat文のかたちは以下のとおりです。

```
repeat
    文 ;
    ⋮
    文
until    式
```

例を示しましょう。上と同じ階乗を求めるプログラムをrepeat～untilを使って書くと次のようになります。

```
var     i, fact : integer
read(i);
repeat
    fact := i;
    i := i-1;
until i <= 0;
write('fact=', fact);
```

このプログラムはrepeat～untilで囲まれた部分を一度実行して，そのあとi＜＝0を評価しそれが真になるまで処理を繰り返します。それでiが1より小さな値が読み込まれたときには，iをそのまま表示します。それからrepeatとuntilのあいだに複数の文が存在するときでもbegin～endで囲む必要はありません。

repeat文のwhileとの違いはwhileは文を実行する前に式を評価してtrueであるなら指定された文を実行します。それに対してrepeatは文を実行したあとに式を評価してfalseであるなら再度文を実行し，式がtrueになったときループを抜けます。それでrepeat文では最低1回は文を実行することになります。

注意しなければならないのは，式がfalseのあいだループするということです。whileは式がtrueのときループするので反対です。ヴィルトはおそらく条件式がtrueのときプログラムの続きに進む，つまりその下の処理を行うと統一しようと考えたのでしょう。Cのdo～whileはtrueのときループするのでCに慣れている方は特に注意してください。

for

for文は回数を指定してループする制御構造です。かたちは次のとおりです。

```
for 制御変数 := 初期値  to 最終値  do 文
```

処理の最初に初期値を制御変数にセットして文を実行し，そのあと制御変数を１ずつ加算しながら文を繰り返し実行し制御変数が最終値に達するまでループします。例を挙げて説明します。

```
for i := 1 to 10 do
    a [i]  := 0;
```

これは，配列ａを初期化する処理を行います。これを実行するとa [1] からa [10] までが0になります。for文を使う際に注意しなければならないのは，for文で使用される制御変数は局所変数でなければならないということです。局所変数というのはそのブロックで宣言された変数のことです。これは制御変数は一時的に処理を制御するために使われるものであるので，非局所変数である必要はなく，プログラムの安全性を高めるためには局所変数にするべきだとの考えに基づいています。それで，関数や手続きの中でfor文を使用するときは必ず使用する関数や手続きのブロックで制御変数を宣言するようにしてください。次の例を見てください。

```
 1: program test(input, output);
 2:
 3: procedure print(n : integer);
 4:   var      i : integer;
 5:       begin
 6:             for i := 1 to 10 do
 7:                   write('*');
 8:       end;
 9:
10:  begin
11:       read(n);
12:       print(n)
13:  end.
```

（注意：行番号は説明のためにつけてありますが実際のプログラムにはありません）

このプログラムは変数ｉがprintという手続きの中で宣言されている局所変数であるのでコンパイルすることができます。それでは，試しに４行目の変数宣言の行を２行目に移してコンパイルしてみると，コンパイラにエラーを指摘されるでしょう。変数ｉが手続きprintの局所変数でなくなったからです。この点for文を使うときは制御変数の宣言は注意してください。

それからfor文でtoの代わりにdowntoを使うこともできます。これを使用すると制御変数を１ずつ減らしながら処理が進められます。

```
for i := 10 downto 1 do
     a [i]  := 0;
```

とすると最初にa [10] :=0が実行され，それからa [9], a [8] の順で処理がなされます。必要に応じて使い分けてください。

goto

PASCALでも一応goto文を使用することができます。でもこれは乱用すると非常にわかりにくいプログラムができてしまいます。PASCALにはせっかく構造化プログラミングを可能とするような制御構造が提供されているのですから，できるだけそれらを使ってプログラミングするようにしましょう。しかし，gotoを使ってもよいと思われるケースもあります。その代表的なケースはエラーが発生したときにただちに処理を終了させなければならない場合です。それらの目的のためにPASCALではgoto文が提供されています。次のようなかたちで記述します。

goto　ラベル

ラベルはgoto文が使用される前に定義されていなければなりません。例を次に示します。

```
label   1000;
var     rc : integer;
          ⋮
    if rc <> 0 then
        goto 1000;
          ⋮
1000:
    end.
```

この例のようにラベルは定数や変数よりも以前に定義します。ラベルは整数でなければなりません。このプログラムでgoto 1000が実行されると制御がendに移りプログラムが終了します。このようにgotoは使用することはできますが，このような特殊な場合を除いて使用しないほうがよいでしょう。

関数と手続き

プログラムを作るときには処理を機能ごとに分解し，それぞれの機能を関数や手続きにまとめて記述するとプログラムを理解しやすいものにすることができます。また何度も使用する汎用性のある処理は，その処理を関数や手続きにしておくと何度も同じ記述をする必要がなくなるので便利です。関数と手続きの違いは，関数は値を返し手続きは処理を行うという点です。それで関数は式の中で使用されますが，手続きは文として使用されます。それでは，詳しく説明していくことにしましょう。

関数

まずPASCALでは関数は使用する前に宣言をしておく必要があります。宣言さえしてしまえばSINやROUNDなどの関数と同じように呼び出すことができます。それでは，最初に宣言の記述法を説明します。関数のかたちは次のとおりです。

function　関数名　仮パラメータリスト :型名；
　　　　　ブロック；

仮パラメータリストは呼び出すときに引数を指定したい場合に記述してください。引数を必要としないときは書く必要はありません。

仮パラメータリストの部分には仮引数の名前と型を指定します。この名前はその関数のブロックで有効です。

型名の部分には関数の戻り値の型を指定します。ただし戻り値の型は単純型またはポインタ型でなければなりません。配列型やレコード型は関数の戻り値にできないのです。

ブロックはラベル定義，型定義，変数宣言，関数および手続き定義とbegin～endで囲まれた文から構成されます。説明のためにひとつ簡単な例を挙げてみましょう。

```
 1: program test(input, output);
 2:   var         a : real;
 3:               b, result : integer;
 4:
 5: function power(x:real, y : integer):real;
 6:   var       i : integer;
 7:           work : real;
 8:           begin
 9:                   work : =1;
10:                   for i : = 1 to y do
11:                           work : =work * x;
12:                   power : =work
13:           end;
14:
15:       begin
16:           read(r, b);
17:           result : =power(a, b);
18:           white('result=',result)
19:     end.
```

(注意：行番号は説明のためにつけてありますが実際のプログラムにはありません)

これはべき乗を求めるプログラムです。5行目から関数powerを宣言しています。powerは最初の引数は実数型で2番目の引数は整数型であり，戻り値は実数型であることを記述しています。6行目から13行目までが関数powerのブロックです。

戻り値は12行目のように関数名に値を代入することによってセットします。呼び出す側は17行目のように書きます。ここでも引数として最初の引数に実数型の式を2番目の引数には整数型の式を与える必要があります。戻り値の型の指定で注意すべきことがひとつあります。この型の指定は必ず型の名前を指定してください。つまり次のように書くことはできません。

```
function test(i:integer) : ^integer;
```

このようなときは次のように型をそれ以前に定義してその型の名前を指定してください。

```
type intp =^integer;
function test(i : integer) : intp;
```

もし上のように^integerのように書き下ろしたかたちで型を指定できたとすればこの関数の戻り値は変数に代入することができないでしょう。すべての変数はこの関数の戻り値とは異なる型を持つことになるからです(先月号の配列型の説明の後半を参照)。

最後にもう一点，PASCALは4種類の引数を提供しています。値引数，変数引数，手続き引数および関数引数の4つです。上の例のプログラムのpowerの引数はすべて値引数です。ここではcall by valueで処理を行っています。値引数と変数引数については7月号の記事で考え方を説明していますので参考にしてください。手続き引数と関数引数とは，引数として関数や手続きを渡せるというものですがあまり使用しない機能ですので今回は説明を割愛します。

手続き

手続きも関数に非常に似ていて，次のように記述します。

function 手続き名　仮パラメータリスト；
　　　　　　　　ブロック；

関数と異なる点は型名がないという点だけです。仮パラメータリストは関数と同じように必要なければ書く必要はありません。このように宣言することによってreadとかwriteといった手続きと同様に使用することができます。呼び出すときはwriteなどと同じように，文として手続き名と必要なら引数を書くだけです。

おまけ

全機種共通システムの記事を書いておられる石上さんが6月号に「急募」として逆ポーランド式から普通の算術式に変換するプログラムを募集しておられたので，私は「再帰を使えば簡単にできるじゃないかあ」と思って，1時間くらいでプログラムを書いて編集部に送ったのです。が，テストもろくにしなかったため完全動作しないまったくお恥ずかしいプログラムだったのです。しかもあとから見てみると，逆ポーランドでなくポーランド式から普通式に変換するものだったのです。

それで，今回ちょうどPurePASCALの連載も制御構造と関数・手続きだったので皆さんの参考にもなるだろうということで，おまけとして「逆ポーランド式を普通式に変換するプログラム」を紹介します。皆さんがPASCALを使ってプログラミングするときの参考にしていただけば嬉しく思います。逆ポーランド式（後置型）とは演算子が式の後ろにくるもので，次のような感じです。

	普通の算術式	逆ポーランド記法
1	a+b	a b +
2	(a−b) * c	a b − c *
3	(a+b) * (c−d)	a b + c d − *
4	a+b*c−d	a b c * + d −

この記述法はスタックを使って演算するときに便利な方法です。数がくるとスタックにプッシュして演算子がくるとスタックの上から2つをとって演算して結果をスタックにプッシュすると最終的な結果がスタックのいちばん上に得られます。それでは，プログラムをコンパイルして実行してみましょう。入力待ち状態になりますので，そこで逆ポーランド式を入力してみてください。普通の算術式が出力されます。

＊　＊　＊

今回は制御構造と関数・手続きについて説明しました。

このようにPASCALは構造化されたプログラムを書くことが可能です。gotoを多用してどこからどこに制御が移るかわからないようなスパゲティプログラムは書かないでください。小手先の高速化はプログラムをわかりにくくすることが多いので，それはやらずにコンパイラ技術が進んできた最近では高速化はコンパイラがやってくれるものと考えたほうがよいでしょう。その点Pure PASCALは最適化をやっていないので，まだまだ改善できる点がたくさん残されていますが……。それでは来月またお会いしましょう。

リスト1

```
==================  POLISH.PAS  ==================
   1: (* 'Polish.pas' *)
   2: (*
   3:    逆ポーランド記法で書かれた式を通常の式の形に直す
   4:            July.1990 by Chack'n
   5: 使い方：
   6:    A>polish          /* 起動 */
   7:    a b + c d - *     /* 入力 */
   8:    (a+b)*(c-d)       /* 出力 */
   9: 注意：
  10:    演算子は＋－＊／の４種類（すべて２項演算子）
  11:    数字と変数名（アルファベット小文字）が使える
  12: *)
  13: program Polish(input, output);
  14:      const
  15:    StackDepth = 100;
  16:      type
  17:    KindOfToken = (IDENT, NUMBER, OP, NONE);
  18:    Str10 = Packed array[1..10]of char;
  19:    TreePtr = ^SyntaxTree;
  20:    SyntaxTree = record
  21:            k:KindOfToken;
  22:            s:Str10;
  23:            left, right:TreePtr
  24:    end;
  25:      var
  26:    cs:char;
  27: (*
  28:    トークンを切り出す
  29: *)
  30:      function GetToken(var IdentString:Str10):KindOfToken;
  31:      var
  32:    c:char;
  33:    i:Integer;
  34:
  35:    procedure Getc(var c:char);     (* 1文字入力 *)
  36:    begin
  37:        if cs = ' ' then
  38:            Read(c)
  39:        else begin
  40:            c := cs;
  41:            cs := ' '
  42:        end
  43:    end;
  44:
  45:    procedure UnGetc(c:char);       (* cを入力へ押し戻す *)
  46:    begin
  47:        cs := c
  48:    end;
  49:
  50:      begin { GetToken }
  51:    for i := 1 to 10 do
  52:        IdentString[i] := ' ';      (* 文字列の初期化<PASCAL
の文字列はいやらしい> *)
  53:    i := 1;
  54:    repeat
  55:        if not Eoln then
  56:            Getc(c)
  57:    until (c <> ' ') or Eoln;       (* スペースを読み飛ばす
*)
  58:    if Eoln then
  59:        GetToken := NONE
  60:    else if c = '.' then
  61:        GetToken := NONE
  62:    else if ('0' <= c) and (c <= '9') then begin
  63:        GetToken := NUMBER; (* 数 *)
  64:        repeat
  65:            IdentString[i] := c;
  66:            i := i + 1;
  67:            Getc(c)
  68:        until (c < '0') or ('9' < c) or Eoln;
  69:        UnGetc(c)   (* 読みすぎた文字を元に戻す *)
  70:    end else if ('a' <= c) and (c <= 'z') then begin
  71:        GetToken := IDENT;  (* 識別子 *)
  72:        repeat
  73:            IdentString[i] := c;
  74:            i := i + 1;
  75:            Getc(c)
  76:        until (c < 'a') or ('z' < c) or Eoln;
  77:        UnGetc(c)   (* 読みすぎた文字を元に戻す *)
  78:    end else begin
  79:        case c of
  80:        '+','-','*','/':
  81:            begin
  82:                GetToken := OP;
  83:                IdentString[1] := c
  84:            end
  85:        end
  86:    end { if }
  87:      end; { GetToken }
  88:
  89: (*
  90:    括弧付けの処理を行いながら、抽象構文木から式を生成する
  91:    括弧付けは演算子の強さを比較する
  92: *)
  93:      procedure Display(t:TreePtr);
  94:      var
  95:    i:integer;
  96:
  97:    procedure WriteIdent(s:Str10);
  98:    var
  99:        i:integer;
 100:    begin
 101:        i := 1;
 102:        While (i <= 10) and (s[i] <> ' ') do begin
 103:            write(s[i]);
 104:            i := i + 1
 105:        end
 106:    end;
 107:
 108:      begin { Display }
 109:    if (t = Nil) or (t^.k <> OP) then
 110:        (* Noting to do *)
 111:    else if (t^.left^.k = OP)       (* 括弧付けの処理 *)
 112:            and ((t^.left^.s[1] = '+') or (t^.left^.s[1] = '
-'))
 113:            and ((t^.s[1] = '*') or (t^.s[1] = '/')) then be
gin
 114:        Write('(');
 115:        Display(t^.left);
 116:        Write(')');
 117:    end else
 118:        Display(t^.left);
 119:    if t = Nil then
 120:        (* Noting to do *)
 121:    else if t^.k = OP then
 122:        write(t^.s[1])
 123:    else
 124:        WriteIdent(t^.s);
 125:    if (t = Nil) or (t^.k <> OP) then
 126:        (* Noting to do *)
 127:    else if (t^.right^.k = OP)      (* 括弧付けの処理 *)
 128:            and ((t^.right^.s[1] = '+') or (t^.right^.s[1] =
'-'))
 129:            and ((t^.s[1] = '*') or (t^.s[1] = '/')) then be
gin
 130:        Write('(');
 131:        Display(t^.right);
 132:        Write(')');
 133:    end else
 134:        Display(t^.right);
 135:    if t <> Nil then
 136:        Dispose(t)
 137:      end; { Display }
 138:
 139: (*
 140:    スタックと２進木を使って逆ポーランド記法から通常の式に直
す
 141: *)
 142:      procedure Polish2Normal;
 143:      var
 144:    TokenStack:array[1..StackDepth]of TreePtr; (* スタックを
使う *)
 145:    sp:integer;
 146:    k:KindOfToken;
 147:    t:TreePtr;
 148:      begin { Polish2Normal }
 149:    sp := 0;
 150:    cs := ' ';
 151:    repeat
 152:        new(t);
 153:        k := GetToken(t^.s);        (* トークンをスタックに
積む *)
 154:        t^.k := k;
 155:        if k <> NONE then begin
 156:            sp := sp + 1;
 157:            t^.k := k;
 158:            TokenStack[sp] := t
 159:        end;
 160:        if k = OP then begin        (* トークンが演算子だっ
たら木を作る *)
 161:            TokenStack[sp]^.left  := TokenStack[sp-2];
 162:            TokenStack[sp]^.right := TokenStack[sp-1];
 163:            TokenStack[sp-2] := TokenStack[sp];
 164:            sp := sp - 2
 165:        end else if k <> NONE then begin
 166:            TokenStack[sp]^.left  := Nil;
 167:            TokenStack[sp]^.right := Nil
 168:        end
 169:    until k = NONE;
 170:    Display(TokenStack[sp]);
 171:    Writeln
 172:      end; { Polish2Normal }
 173:
 174: begin { main }
 175:      Polish2Normal
 176: end. { main }
```

BASIC

X-BASICプログラミング調理実習(14)

ファイルの魔術師fseek関数

Izumi Daisuke 泉 大介

ファイル処理の基本を理解したところで，ちょっとした高度な技をご紹介しましょう。一般にランダムアクセスが必要なデータベースなどをX-BASICで作るにはある命令を使うことが必要になります。それが今回登場するfseek関数というわけです。

先月はファイル処理の入門編としてファイルを扱ううえでの約束ごとを中心にお送りしました。ファイルを扱うには，まずファイルをオープンし，データの読み書きを行ったあと，最後にファイルをクローズする必要があること。ファイルのオープン時にはデータを読み出すのか，書き込むのか，読み書きするのか，ファイルを新しく作るのかを指定する必要があること。この指定のことをモードということ。思い出していただけましたか。

ファイルを扱ううえで大切なものとしてファイル番号がありました。これはファイルと1対1に対応する整数で，データのやり取りやファイルをクローズするときにファイル名の代わりに使います。つまり，いったんファイルをオープンすれば，以後の作業はすべてファイル番号で行うということです。

先月はこれらのことを踏まえ，ファイル処理の実例としてYETのスコアファイル関連の処理を行いました。トップ10の中から同一人物を削除する，2つのyetscoファイルを1つにまとめる，という2つの処理を行うプログラムです。どうでしょうか。役に立っていますか？

シーケンシャルかランダムか

では基礎事項を踏まえ，ちょっと高度なファイル処理に挑戦してみましょう。といっても命令としては先月紹介したファイル処理関係の命令にたった1つ命令が加わるだけです。

先月扱ったBASICのプログラムファイルのことをちょっと思い出してください。ファイルの先頭には空白に続く行番号があり，その後ろに命令が埋め込んでありました。このようにファイルの先頭からデータを順次読み込んでいって初めて意味のあるデータを取り出せるというのは，BASICのプログラムファイルに限ったことではありません。ワープロの文書ファイルもそうですし，エディタで作成するCのプログラムなどもそうです。このような性格をもったファイルをシーケンシャルファイルといいます[1)]。

ファイルにはもうひとつ，ランダムアクセスファイルと呼ばれるものがあります。シーケンシャルファイルがファイルの先頭から順次データを処理していくファイルなのに対し，こちらはファイル内の任意の場所から直接データを取り出すことができるという特徴を持っています。

例を挙げて説明しましょう。ここに毎日午前6時の気温を1年間にわたって記録したファイルがあるとします。8月18日の気温を調べようとすると，シーケンシャルファイルでは229個のデータを読み込んでは捨て，やっと230個目に目的のデータに到達するという手順になります。ところがランダムアクセスファイルでは，230個目のデータを一発で取り出すことができるのです。したがってランダムアクセスファイルはデータベースのように任意のデータを読み出す必要のあるファイルに向いているといえるでしょう。

さて，われらがX-BASICでは，ファイルをランダムアクセスファイルとしてオープンするなどという必要はなくなっています。基本的にX-BASICではシーケンシャルファイルしか扱わないのです。げげっ！　それじゃァ229個のデータを読み捨てるという思いっきり時間のかかりそうな作業が必要なのか……と思われるかもしれませんね。でも大丈夫。落胆することはありません。X-BASICは確かにシーケンシャルファイルしか扱いませんが，その代わりデータを読み書きする位置を自分で好きなように変えることができるのです。

●位置ズラしの術

データを読み出す，書き込む位置を自由に変えられる？　そう，これが冒頭でほのめかした，追加される1つの関数の役割です。

```
Welcome to X-BASIC
^
```

というデータの入ったファイルがあるとしましょう。^印はこれからデータを読み出そうとする位置を表しています。ここから1文字ずつデータを読み

1) データの並び（シーケンス）をそのまま先頭から順に読み込んで処理するのでこの名前がついています。

出していくと，

```
Welcome to X-BASIC
 ^

Welcome to X-BASIC
  ^

Welcome to X-BASIC
   ^
```

と読み出す位置が1つひとつ後ろへズレていき，W, e, lとデータが読み出されていきます。さてここでデータを読み出す位置を3つ後ろへズラしてみることにしましょう。

```
Welcome to X-BASIC
      ^
```

ですね。ここで1文字読み出すと，読み出されるのはeになります。間の「com」を飛ばしてしまったわけですね。

上では1文字ずつデータを読み出しましたが，考え方は数値を収めたファイルでも同じです。

```
21 23 25 27 29 28 26 24 22 ……
^
```

と毎日の気温を保存したファイルがあるとします。ここで数値データを1つ読み込めば21度という気温を読み出すことができます。

```
21 23 25 27 29 28 26 24 22 ……
   ^
```

読み出す位置をここから4つ後ろへズラせば，次は28度という気温が読み出せます。ファイルの先頭を1月1日だとすると，そこから数値データ230個分後ろへ読み出す位置を動かせば，229個のデータを無駄読みしなくても一発で8月18日の気温を取り出せるという寸法です。

●fseek関数の使い方

位置ズラしはfseek関数によって行います。この関数は，

fseek(fileNo, offset, mode)
　fileNo：ファイル番号
　offset：オフセット
　mode：モード

という書式で使います。オフセットというのはいくつズラすかというズラし量の設定で，整数で指定します。モードも同じく整数で，現在の読み出し位置からズラすのなら1を指定します。モードはこのほかにファイルの先頭から（mode=0）とファイルの最後から（mode=2）を指定できるようになっています。

では早速実験してみましょう。まずはデータを入れるファイルの作成です。testという名前にしましょう。

```
int fp
```

```
fp = fopen( "test", "c" )
```

ですね。これでtestというファイルが作れました。お次はfseek関数です。1つだけズラしてみることにしましょう。

```
fseek( fp, 1, 1 )
```

です。さて，どうなりましたか？　「指定の位置にはシークできません」とエラーになったはずです。testファイルはいま作成したばかりですから，データはなにも入っていません。データのないところへ位置をズラそうとしたためにエラーになったわけです。

データがないといっているのですから，作ってやることにしましょう。

```
int data( 100 )
fwrite( data, 100, fp )
fseek( fp, 0, 0 )
```

整数型の配列を用意し（この時点ですべて0が入った配列が作成される），そのデータを100個保存してみました。最後にfseek関数でファイルの先頭に位置を動かします。今度はどうですか？ fseek関数はちゃんと動いたでしょうか。

fseek関数が動作したことを確かめたなら，次にどこまで位置を動かせるかも試してみましょう。これは，

```
int i
for i=0 to 999 : fseek(fp, i, 0) : next
```

とやってみればすぐにわかります。データのないところへは動かせないのですから，エラーになったらiを表示してみればどこまで動いたのかすぐにわかるはずです。実行してみましょう。

ピッ！　エラーが発生したときのiの値は，ナント401になっています。100個しかデータはないはずなのに400個のデータ分だけ動かせたということです。なぜこんなことになるのでしょうか。

●データの「大きさ」

これは,「整数は1文字分のデータでは表現できない」というのが原因です。X-BASICでは文字型をchar型として表現します。char型は0〜255の範囲の整数で，ASCIIコードの範囲と同じです[2)]。ところが整数（int型）は±21億という大きな数を扱えるのです。とても1バイト分のデータで表現できる大きさではありません。X-BASICで扱うデータ型とその大きさは,

char：1バイト　□

int　：4バイト　□□□□

float：8バイト　□□□□□□□□

となっています。後ろの四角はデータの大きさを視覚的に表現してみたものです。

上の例でtestファイルに収めた整数（int）100個分は，文字（char）でいうなら4倍の400個分，すなわち400バイトに相当するのです。そしてfseek関数は，オフセットをバイト単位で指定するようになっています。testファイル内で読み出し位置をズラしていったとき，401までズラしたところでエラーが出たのはこういうわけだったのです。

●好きな位置でデータを読み書きしてみる

整数を収めるには4バイト必要だということがわかってしまえば，自分の好きな位置からデータを読み出すのは簡単でしょう。

fseek(fp, 0, 0)

でファイルの先頭に読み書き位置を戻したら,

for i=0 to 99：data (i)=i：next

fwrite (data, 100, fp)

を実行してください。配列のi番目にiという数値を入れ，testファイルに収めたわけです。これを使って自分の好きな位置からデータを取り出してみることにします。

数値データを取り出すには数値型の配列を使いfread関数を使うしか手がありませんから，まずデータ取り出し用に配列を1つ用意します。

int a(0)

testファイルの50番目に入っているデータを取り出してみましょう。これは,

fseek(fp, 50 * 4, 0)

fread(a, 1, fp)

でOKですね。整数型は1データ4バイトだということを忘れないでください。

print a(0)

として取り出したデータを表示してみると，見事50が表示されます。

違う型のデータを混在

皆さんはX-BASICが扱うそれぞれのデータが何バイトで表現されているかがわかり，データ書き込み読み出し位置を自由に設定できるようにもなりました。もういろいろな型のデータをファイル内に混在させることも簡単にできるはずです。簡単な例として，あるクラスの生徒名と身長・体重を収めたデータファイルを作ってみましょう。

名前は漢字5文字分（10バイト）もあれば十分でしょうが，余裕を見て20バイト使うことにしましょう。身長はmm単位の整数，体重はg単位の整数で表現すれば十分そうです。それぞれ整数ですから4＋4＝8バイト。名前と併せると1人分のデータは28バイトで表現されることになります。これがクラスの人数分続くわけですね。イメージは図1のようになります。

では簡単なプログラムを使って実際にデータをファイルに書き込んでみることにしましょう。リスト1です。

プログラムの先頭でデータ入力に必要な変数と

2) 0〜255は16進2桁で表現できる数(1バイト：00_H〜FF_H)。コンピュータではこれを基本単位としてデータのサイズを表現することが少なくありません。

図1　身長・体重表の表現

リスト1　簡単なデータベース その1

```
 10 str   pname[20]                       /* 名前入力用
 20 int   height                          /* 身長入力用
 30 int   weight                          /* 体重入力用
 40 dim   buf(0)                          /* データ登録用配列
 50 char dummy(28)                        /* データベース作成用ダミー
 60 /*
 70 int dataNo                            /* データの個数を保持
 80 int fp
 90 int i
100 /*
110 fp = fopen( "data", "c" )             /* データベース作成
120 for i=1 to 50
130   fwrite( dummy, 28, fp )             /* 50人分の領域を作成
140 next
150 fseek( fp, 0, 0 )                     /* 先頭に戻す
160 /*
170 for dataNo = 0 to 49
180   pname = ""                          /* 名前をヌルにしてから
190   input "名前：",pname                /* 名前を入力
200   if pname = "" then break            /* ヌルなら終了する
210   input "身長：",height
220   input "体重：",weight
230   print
240   /*
250   fwrites( pname, fp )                /* 名前を登録する
260   fseek( fp, 28*dataNo + 20, 0 )      /* 身長を登録する位置まで移動
270   buf(0) = height
280   fwrite( buf, 1, fp )                /* 身長を登録
290   buf(0) = weight
300   fwrite( buf, 1, fp )                /* 体重を登録
320 next
330 fclose( fp )
```

データ書き込みに必要な変数を宣言しています。40行で宣言しているbuf配列は数値データをファイルに書き込むのに使用。50行のdummy配列は最初にデータベースを作るのに使用しています。

プログラムは実に単純で，最初はデータベースファイルの作成から始まります。オープンしたらdummy配列を50回書き込んで，50人分のデータを入れる領域の確保です。データがないところへは読み書き位置を移動できないことを思い出してください。領域の確保が終わったらfseek関数で読み書き位置をファイルの先頭に戻しておきます（以上150行まで)。

170行から始まるfor～nextループが実際にデータ書き込みを行う部分です。名前，身長，体重をinput命令で読み込み，250行以降で順にファイルに書き込んでいきます。ここで注意すべきことは，fwrites関数で文字列をファイルに書き込んだ場合には，書き込んだ文字数だけしか読み書き位置が移動しないということです。pname配列が20文字分用意してあるからといって20文字書き込むというわけではありません[3)]。そこで260行でfseek関数を使い，名前を書き込んだあと次の身長を書き込む位置まで読み書き位置を動かす必要があるのです。

for～nextはこの作業を名前がヌルになるか50名のデータを入力し終わるまで繰り返します。つまり，名前を入力するときに単にリターンキーだけを押せば入力途中で終了できるようになっています。このためには190行を忘れてはいけません。input命令で値を入力せずに単にリターンキーを押すと，前に入っていた値がそのまま更新されずに採用されてしまいます。前回が「泉　大介」だったなら，単にリターンキーを押しただけではpname配列には“泉　大介”がセットされたままになってしまうのです。

データを修正できるように

このままではデータを入力することしかできませんから，せめてデータの表示・修正くらいはできるように変更してみることにしましょう。いろいろな方法が考えられるでしょうが，ここでは最も簡単なメニュー選択式を採用することにします。

プログラムの動作を考えておくことにしましょうか。プログラムを実行すると，

1)　データ表示
2)　データ修正
3)　データ入力
4)　終了

というメニューが画面に表示されることにします。1)～3)のメニューを選択すると，さらにデータ番号を尋ねてくるようにしましょう。長いデータを常に最初から表示というのではうんざりですし，データ修正では修正したいデータを指定しなければなりませんからね。データ入力も指定した番号のデータ以降に順に追加していけるというのがいいでしょう。

このデータ番号を元にfseekして目的の場所へ読み書き位置を移動し，リスト1のようなプログラムでデータの入力・修正を行うようにすればいいでしょう。表示も同様です。では実際のプログラムを見ていただくことにしましょうか。リスト2です。

まず最初にやるべきことは，すでにdataというファイルが存在しているかどうかを調べることです。fopen関数はファイルが存在しなければファイル番号の代わりに－1を返しますから，

```
fp=fopen( "data", "rw" )
```

とし，返されたfpが－1かどうかを調べればOKのように思われます。しかしそうは問屋が卸しません。ダイレクトモードで，

```
print fopen( "abcde", "rw" )
```

などと存在しないファイルをオープンしてみればすぐにわかります。エラーになりますね。ではどうするかというと，

```
error off
```

という便利な命令が用意されているのです。これはX-BASICの外部関数（fopenもそのひとつ）でエラーが発生しても，それを無視するという命令です。この命令はプログラム中でしか使用できません。リスト2ではこれを使ってdataファイルが存在するかどうかをチェックしています。まずdataファイルを“rw”モードでオープンしてみてfpが－1でなければそれでよし。－1ならば改めて“c”モードでオープンするというわけです。ファイルが存在するかどうかを調べる定石として覚えておいてください。

dataファイルがオープンできたらメニューを表示し，メニュー選択に入ります。そして押されたキーによってswitchで処理を分ける。と，こんな調子でプログラムは進んでいきます。それぞれの処理が終わったら再びメニューに帰ってくるようにしたいので，メニュー選択・処理分岐部分はwhile～endwhileのループにしておくのがいいでしょう。そうそう，データ番号を入力してもらうのを忘れてはいけません。switchで処理を分ける前に入力してもらうことにしましょう。

データ追加はリスト1の焼き直しで，addDataという関数名で定義しました。変更点は，データを1つだけ入力するのかファイルの最後まで（maxDataという変数に定義している）入力を続けるのかを判定するようにしたことです。修正用のプログラムを簡単に作成するための変更です。これはすぐでしょ

3)　一見不便なようですが，テキストファイルを扱うときにはこうでないと都合が悪いのです。1行書き込むたびに256文字分のスペースがとられてしまう……なんて事態を想像してみてください。

うから，残るはデータ表示と修正だけですね。

データの表示はdispDataという関数が受け持っています。入力されたデータ番号に従って読み書き位置を移動し，名前，身長，体重の３つのデータを１行に表示していきます。この作業をmaxDataまで繰り返せばファイル表示は終了ですね。表示されたデータを確認できるよう，最後にキー入力を待ってから終わることにしています。データの１行表示はlineDataという関数に任せました。これも修正処理を簡単に済ませたかったからです。

そのlineData関数はファイルにデータを書き込むときとまったく逆のことを行っているだけです。まず名前を読み出し，続いて身長と体重をいったんbuf配列に読み出しておいて表示します。久々に，

locate 30, csrlin

というのが登場していますね[4]。

最後に残ったデータの修正ですが，addData関数を変更しデータの１行表示をlineData関数に任せたおかげで，実に簡単になりました。editData関数が修正を受け持つ関数なのですが，目的のデータに読み書き位置を移動し，lineData関数でまずはデータ内容を表示，続いてaddData関数を呼び出して新しいデータを入力してもらうだけです。

これでプログラムは完成です。今回はちょっと高度なファイル処理ということで，異なったデータ型をひとつのファイルに収める方法について紹介しました。プログラムはこれぞサンプルといわんばかりの簡潔さですが，基本は理解いただけたのではないかと思います。自分が作ろうとするデータベースの構造にあわせてプログラムを変更してみてください。

来月は今回のランダムアクセスファイルを踏まえ，さらにこれまでの知識を総動員してカード型データベースに挑戦してみたいと思います。ランダムファイルは今回紹介したとおりです。あとはいかに使いやすくするか，見栄えをよくするかといったところでしょうか。X-BASIC総集編として楽しみにしてください。

4)　csrlinはX-BASICが用意しているシステム変数で，現在カーソルのあるＹ座標を保持している変数です。表示するＸ座標だけを変更したいというときに「カーソル行は今のまま」と指示するのに使います。逆に「Ｘ座標は今のまま」でＹ座標だけ変えたいという場合には，posというシステム変数を使い，

locate pos, 10

のようにします。

お願い

このX-BASICプログラミング調理実習も14回を重ね，基本的な題材もほぼ一巡した感があります。もちろん，毎月毎月新しい読者がOh! Xを読み始めるわけですから，一通りやればいいというものでもありませんね。やはりテーマを変えながらも必要なことは何度でも繰り返しやっていかなければならないでしょう。

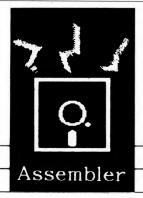
さて，今後への参考のためにお聞きしたいのですが，

1)　コマンドモードを使えますか？

2)　BASICでなにかをやってみたいと思いますか？

というわけで，アンケートハガキのすみっこにでも1)Ｙ 2)Ｎ とか記入してもらえると助かります。もちろんご意見も大歓迎。よろしくお願いします。

リスト2　簡単なデータベース その2

```
  10 str  pname[20]                       /* 名前入力用
  20 int  height                          /* 身長入力用
  30 int  weight                          /* 体重入力用
  40 int  maxData = 50                    /* 最大データ数
  50 dim  buf(0)                          /* データ登録用配列
  60 char dummy(28)                       /* データベース作成用ダミー
  70 /*
  80 int  dataNo                          /* データの個数を保持
  90 int  fp
 100 /*
 110 int  i
 120 str  ky
 130 /*
 140 error off                            /* エラーで止まらない
 150 fp = fopen( "data", "rw" )           /* dataは既にあるか？
 160 error on
 170 if fp = -1 then {
 180   fp = fopen( "data", "c" )          /* なければ作成
 190   for i=1 to maxData
 200     fwrite( dummy, 28, fp )          /* 50人分の領域を作成
 210   next
 220   fseek( fp, 0, 0 )                  /* 先頭に戻す
 230 }
 240 while 1
 250   cls                                /* メニュー表示
 260   print "1)  データ表示"
 270   print "2)  データ修正"
 280   print "3)  データ入力"
 290   print "4)  終了"
 300   print
 310   print ">";
 320   ky=inkey$                                    /* 選択
 330   if instr( 1, "1234", ky )=0 then continue    /* 選択間違い
 340   if ky="4" then break
 350   input "データ番号：",dataNo        /* データ番号入力
 360   cls
 370   switch ky
 380     case "1"
 390       dispData()
 400       break
 410     case "2"
 420       editData()
 430       break
 440     case "3"
 450       addData( 0 )
 460       break
 470   endswitch
 480 endwhile
 490 fclose( fp )
 500 end
 510 /*
 520 func dispData()
 530   int i
 540   str ky
 550   fseek( fp, dataNo*28, 0 )
 560   for i=dataNo to maxData-1
 570     lineData( i )                    /* データ表示
 580     ky = inkey$(0)
 590     if asc( ky ) = 27 then break     /* ESCが押された
 600   next
 610   print
 620   input "キーを押してください  ",ky
 630 endfunc
 640 /*
 650 func lineData( no )
 660   print using "##@";no," : ";        /* データ番号表示
 670   freads( pname, fp )
 680   print pname;                       /* 名前表示
 690   fseek( fp, 28*no + 20, 0 )
 700   fread( buf, 1, fp )
 710   locate 30, csrlin
 720   print buf(0),                      /* 身長表示
 730   fread( buf, 1, fp )
 740   print buf(0)                       /* 体重表示
 750 endfunc
 760 /*
 770 func editData()
 780   fseek( fp, dataNo*28, 0 )
 790   lineData( dataNo )                 /* データ表示
 800   addData( 1 )                       /* データ修正
 810 endfunc
 820 /*
 830 func addData( mode )
 840   int i
 850   int endNum
 860   /*
 870   if ( mode ) then {                 /* modeが1なら1つだけ入力
 880     endNum = dataNo
 890   } else {
 900     endNum = maxData-1
 910   }
 920   /*
 930   fseek( fp, dataNo*28, 0 )          /* 書き込み位置まで移動
 940   for i = dataNo to endNum
 950     pname = ""                       /* 名前をヌルにしてから
 960     print "#";i
 970     input "名前：",pname             /* 名前を入力
 980     if pname = "" then break         /* ヌルなら終了する
 990     input "身長：",height
1000     input "体重：",weight
1010     print
1020     /*
1030     fwrites( pname, fp )             /* 名前を登録する
1040     fseek( fp, 28*i + 20, 0 )        /* 身長を登録する位置まで移動
1050     buf(0) = height
1060     fwrite( buf, 1, fp )             /* 身長を登録
1070     buf(0) = weight
1080     fwrite( buf, 1, fp )             /* 体重を登録
1090   next
1100 endfunc
```

直接グラフィックを操作する

Murata Toshiyuki 村田 敏幸

皆さんのなかにも，あまりDOSコールやIOCSコールのお世話になってばかりではどうもマシン語プログラミングらしくないと欲求不満になっている人はいませんか。今回はX68000のグラフィック画面を直接アクセスしてプログラムを書いてみることにしましょう。

今回は“単純なわりにはマシン語している気分になれるグラフィック”というテーマで押してみる。IOCSコールのグラフィック描画機能については前回軽く触れてある。パラメータを与えるだけで図形が描けるという手軽さ・簡便さが最大の長所だった。反面，機能や速度の点ではかなり不満があるというのも事実だ。点や線，円といった最低限の図形しか描けないし，描画モードの指定もできないし[1]，描画速度もそれほど速いというわけではない[2]。もうひとつ，実はこれがいちばん大きな不満なのだが，“どうもマシン語っぽくなくて”面白くない。そういうわけで，IOCSを使わずに直接グラフィック画面に書き込みを行うことを考えてみたい。

グラフィック画面とアドレス計算

グラフィック画面の実体はグラフィックVRAM（Video RAM）とかG-RAMと呼ばれるメモリであり，このメモリにデータを書き込むことがグラフィック描画にあたる。

X68000のG-RAMは（正しい用語かどうかは知らないが俗にいう）垂直型で，メモリの1ワードと画面上の1ドットが1対1に対応している。点を打ちたければ，座標から算出したG-RAM上のアドレスにパレットコードを書き込めばよい。X1やPC-9801など多くのマシンで採用されている（いわゆる）水平型のG-RAMでは単色の画面（プレーン）を複数重ね合わせてカラー表示を行う構成なので1つ点を打つのにもプレーンの枚数分の書き込み操作を行わなければならず，しかも横方向の8ドットが1バイトにまとまっているためにビット単位の操作も必要になる[3]。扱いやすさという点では断然垂直型に軍配が上がるし，速度の点でも多くの場合，垂直型のほうが有利だ。

図1～3にX68000のG-RAMの構造を示す。図1は画面モード別のメモリマップ，図2は実画面の大きさ別の画面内のワードの並び方，図3は色モード別のワード内部の構成を表している。特徴的なのは，画面モードに応じてG-RAMのメモリ空間上の大きさが変化する点だろう。最大では2Mバイトもの空間を占める。X68000のG-RAMは512Kバイトしかないからつじつまが合わないように見えるが，図3を見てもらえればわかるように，256色や16色モードのときは各ワードの上位ビットが空いており，結局，256色モードのときは，

8ビット×512ドット×512ドット×2画面
=512Kバイト

16色モードのときは，

4ビット×512ドット×512ドット×4画面
=512Kバイト

もしくは，

4ビット×1024ドット×1024ドット×1画面
=512Kバイト

となり，メモリの総容量が変わるわけではない。

では，図を参考に，xy座標からG-RAM上のアドレスを計算する手順・プログラムを考えてみよう。図2を見ればG-RAM上のメモリは画面の左から右方向にアドレスが増加し，右端で折り返して下のラインの左端につながるという形をしていることがわかる。隣り合ったドットのアドレスの差は，左右が2バイト，上下が2048（実画面1024×1024ドットモード時）または1024（512×512ドットモード時）バイトという定数になる。これから，ページ0の座標(x,y)に対応するG-RAM上のアドレスは，実画面1024×1024ドットモードのとき，

$C00000_H + x \times 2 + y \times 2048$

実画面512×512ドットモードのとき，

$C00000_H + x \times 2 + y \times 1024$

と導ける。ページ1以降の場合は先頭アドレスが80000_Hバイトずつずれるだけだから，その分を足してやればよい。

リスト1が実際にアドレスを計算するプログラムの一例だ。座標からのアドレス計算はこれからもたびたび使うことが予想されるので，独立したサブルーチンとし，関連サブルーチンと一緒にモジュール化してある。

サブルーチンgramadrはd0.wにx座標，d1.wにy座標を入れて呼び出すと，対応するG-RAM上の

1) X68000が初めてのマシンだという人のために補足すると，多くのホビーマシンのBASICではグラフィック描画時に描画を行う点の色と描画色のあいだでさまざまな演算を行う機能が用意されている。XOR（68000風にいえばEOR）モードなどは画面上に一時的に枠を描くときなんかに重宝したものだ。

2) IOCSコール呼び出しそのもののオーバーヘッドに加えて，画面モードに応じた処理の振り分けや，厳格なエラーチェック，速度よりもメモリ効率を優先したコーディングなどが速度を落とす要因になっている。汎用性を追求すると，どうしたって速度が犠牲になるものだ。

3) 逆にいえば，単色に限れば1バイトの書き込みで横8ドット分が描けるのが水平型の利点といえる。

4) 2048は2の11乗だから11ビット左シフトすれば2048倍したことになる。同様に1024は2の10乗だから1024倍するには10ビット左シフトすればよい。

アドレスをa0に返す。y座標を2048倍または1024倍する部分は遅い乗算命令ではなく，ビットシフトに置き換えて計算している[4]。何ビットシフトするかはインクルードしているリスト2のGCONST.H内で定数定義してある。

また，描画ページの先頭アドレスは31行以下の描画ページを設定するサブルーチンapage内であらかじめ計算し，ワークgbaseにしまっておくようにした。これにより，わずかながらアドレス計算時の手間を省くことができる。apageの中ではワークgbaseにアドレスをセットするだけではなく，IOCSコールAPAGEを呼び出すことでIOCS側にも描画ページの変更があったことを伝えている。これは，gramadr(を利用してこれから作るサブルーチン)とIOCSのグラフィック描画機能を混在して使用する場合に備えてのことだ。

あと，特殊な用途にも対応できるように，ワークgbaseの値を直接変更するsetgbase，gbaseの値を読み出すgetgbaseの2つのサブルーチンを用意してある。gbaseを.xdefで外部定義せずにサブルーチンを通して間接的に読み書きするようにしたのには，モジュールの独立性を保つ意味がある。

●1024/512ドットモードへの対応

ここで，リスト1は実画面が1024ドットの場合と512ドットの場合のどちらでも使えるようになっており，条件つきアセンブルを利用して，アセンブル時にどちらのモード用かを決定する。

```
    as gramadr
```

のようにふつうにアセンブルすると512ドットモード用，そして，

```
    as gramadr /d_1024
```

というように“_1024”というシンボルを定義すると1024ドットモード用のオブジェクトが生成される。サブルーチンの中でモードに応じて処理を振り分けることもできたし，なんならモード別に2つのサブルーチンを用意してもよかったのだが，1本のプログラムで複数の画面モードを使うことはあまりないだろうという判断から，こういう形式にしてみた。

条件つきアセンブルされる部分はリスト1本体ではなく，インクルードしているリスト2中に隠れている。1～14行が条件つきでアセンブルされる部分だ。

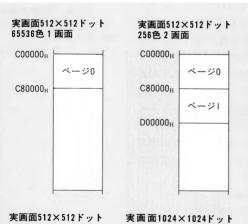
```
    .ifdef    _1024
    (シンボル_1024が定義されているときアセンブルされる部分)
    .else
    (シンボル_1024が定義されていないときアセンブルされる部分)
    .endif
```

のような形になっているのがわかると思う。

AS.Xにはこのほかにも数式の比較結果に応じて条件つきアセンブルする機能が用意されている。こっちのほうはマクロ定義の中で，

```
    macro DOS2    callno, size
        .dc.w     callno
    .if size>8
        lea.l     size(sp),sp
    .else
```

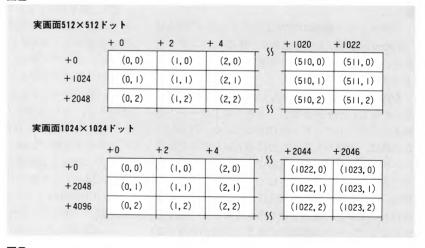
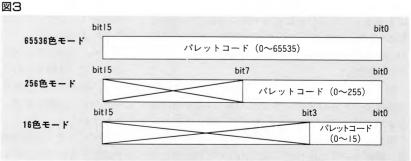
図1

図2

実画面512×512ドット

	+0	+2	+4	...	+1020	+1022
+0	(0,0)	(1,0)	(2,0)		(510,0)	(511,0)
+1024	(0,1)	(1,1)	(2,1)		(510,1)	(511,1)
+2048	(0,2)	(1,2)	(2,2)		(510,2)	(511,2)

実画面1024×1024ドット

	+0	+2	+4	...	+2044	+2046
+0	(0,0)	(1,0)	(2,0)		(1022,0)	(1023,0)
+2048	(0,1)	(1,1)	(2,1)		(1022,1)	(1023,1)
+4096	(0,2)	(1,2)	(2,2)		(1022,2)	(1023,2)

図3

```
        .if size>0
                addq.l      #size,sp
        .endif
        .endif
                .endm
```

のような感じで使ったりする。このマクロはDOSコールの呼び出しとその後のスタックポインタ補正をまとめたもので，

```
        DOS2        _PUTCHAR,2
        DOS2        _WRITE,10
        DOS2        _EXIT,0
```

のように利用することを想定している。条件つきアセンブルを利用して，第2パラメータの値に応じてaddq.lかlea.lの適切なほうを使ってスタックポインタを補正する。上の例はそれぞれ，

```
        .dc.w       _PUTCHAR
        addq.l      #2,sp

        .dc.w       _WRITE
        lea.l       10(sp),sp

        .dc.w       _EXIT
```

のように展開されることになる。

●画面への書き込み

座標からG-RAM上のアドレスを得るサブルーチンが用意できたところで，グラフィック画面に点を1個打ってみる。これはもう簡単で，

```
        move.w      x座標,d0
        move.w      y座標,d1
        bsr         gramadr
        move.w      パレットコード,(a0)
```

とやれば，望みの位置に望みの色で点が打てるはずだ。ただし，X68000のG-RAMはユーザーモードからはアクセスできないスーパーバイザ空間に割りつけられているから，あらかじめスーパーバイザモー

リスト1　GRAMADR.S

```
 1:         .include        iocscall.mac
 2:         .include        gconst.h
 3: *
 4:         .xdef   gramadr
 5:         .xdef   setgbase
 6:         .xdef   getgbase
 7:         .xdef   apage
 8: *
 9:         .text
10: *
11: *       (d0.w,d1.w) → a0 = adr
12: *
13: gramadr:
14:         movem.l d0-d2,-(sp)
15:
16:         ext.l   d0                      *座標をワードから
17:         ext.l   d1                      *  ロングワードにしておく
18:
19:         moveq.l #GSFTCTR,d2             *yを2048または1024倍する
20:         asl.l   d2,d1                   *
21:         add.l   d0,d1                   *それにxの2倍を足す
22:         add.l   d0,d1                   *
23:         add.l   gbase(pc),d1            *G-RAMの先頭アドレスを足す
24:         movea.l d1,a0                   *結果はa0で返す
25:
26:         movem.l (sp)+,d0-d2
27:         rts
28: *
29: *       d1.w = ページ番号(0～3)
30: *
31: apage:
32:         IOCS    _APAGE          *IOCSにもページ番号を伝える
33:         tst.l   d0              *エラー？
34:         bmi     apage0          *  そうなら終了
35:
36:         move.w  d1,d0           *d0.w=ページ番号
37:         lsl.w   #19-16,d0       *ページ番号を
38:         swap.w  d0              *  $80000倍する
39:         clr.w   d0              *
40:         addi.l  #GPAGE0,d0      *G-RAMの先頭アドレスを足す
41:         move.l  d0,gbase        *ワークにしまう
42:
43:         moveq.l #0,d0   *APAGEの戻り値を復帰する
44: apage0: rts
45: *
46: setgbase:
47:         move.l  a0,gbase
48:         rts
49: *
50: getgbase:
51:         move.l  gbase(pc),a0
52:         rts
53: *
54: gbase:  .dc.l   GPAGE0
55:
56:         .end
```

リスト2　GCONST.H

```
 1: .ifdef 、_1024
 2:                 *実画面1024x1024
 3:
 4: GNPIXEL equ     1024    *1ラインのピクセル数
 5: GNBYTE  equ     2048    *1ラインのバイト数
 6: GSFTCTR equ     11      *GNBYTE倍するためのシフト回数
 7: .else
 8:                 *実画面512x512
 9:
10: GNPIXEL equ     512
11: GNBYTE  equ     1024
12: GSFTCTR equ     10
13:
14: .endif
15:
16: GPAGE0  equ     $c00000 *グラフィックページ0先頭アドレス
17: GPAGE1  equ     $c80000 *                  1
18: GPAGE2  equ     $d00000 *                  2
19: GPAGE3  equ     $d80000 *                  3
20:
21: GPALET  equ     $e82000 *グラフィックパレット先頭アドレス
```

リスト3　PSETTEST.S

```
 1:         .include        doscall.mac
 2:         .include        iocscall.mac
 3:         .include        const.h
 4: *
 5:         .xref   gramadr
 6: *
 7: CSCREEN         equ     16
 8: DOS_GL3         equ     5
 9: *
10:         .text
11:         .even
12: *
13: ent:
14:         lea.l   mysp(pc),sp             *spを初期化する
15:
16:         move.w  #DOS_GL3,-(sp)          *画面を512x512,65536色に
17:         move.w  #CSCREEN,-(sp)          *  初期化
18:         DOS     _CONCTRL                *
19:         addq.l  #4,sp                   *
20:
21:         clr.l   -(sp)                   *スーパーバイザモードへ
22:         DOS     _SUPER                  *  移行
23:         move.l  d0,(sp)                 *現在のsspを待避
24:
25:         move.w  #256,d0                 *x=256
26:         move.w  d0,d1                   *y=256
27:         bsr     gramadr                 *G-RAM上のアドレスを得る
28:         move.w  #65535,(a0)             *点を打つ
29:
30:         DOS     _SUPER                  *
31:         addq.l  #4,sp                   *ユーザーモードへ復帰
32:
33:         DOS     _EXIT                   *終了
34: *
35:         .stack
36:         .even
37: *
38: mystack:
39:         .ds.l   256
40: mysp:
41:         .end    ent
```

ドに移行しておく必要がある。

リスト3に簡単なサンプルを示しておこう。当然，リスト1とリンクして使う。65536色モードにし，画面中央の座標（256,256）に白で1個点を打つプログラムだ。サブルーチンgramadrが正常に動作しているかどうかテストするつもりで走らせてみてもらいたい。

ボックスを塗り潰す関数

点が打てるところまで漕ぎ着けた。理屈のうえでは点が打てればどんな図形でも描けるわけで，あとは描画アルゴリズムだけの問題となる。試しにX-BASICのグラフィック描画関数のなかで（psetを除けば）いちばん簡単に実現できそうなfill関数の相当品を作ってみることにする。

fillの描画アルゴリズムについては説明するまでもないだろう。x座標を1ずつ増やしながら順に点を打っていく処理を長方形の高さの回数だけ繰り返せばよい。プログラムは単純な2重ループになる。あっという間にできたプログラムがリスト4。サブルーチンだけなので，動作試験にはリスト5のテスト用プログラムを利用してほしい。

パラメータは適当なメモリ領域に左上の座標，右下の座標，パレットコードの順に格納し（IOCSコールFILLと同じ形式），その先頭アドレスをスタックに積んで渡すように作ってある。なお，リスト4もGCONST.Hをインクルードしているので，アセンブル時にシンボル_1024が定義されていれば実画面1024ドットモード用，定義されていなければ512ドットモード用のオブジェクトが得られる。今月の以下のプログラムもこのノリだ。

順に見ていこう。23行まででパラメータが，

d0 ＝　左上x座標
d1 ＝　左上y座標
d2 ＝　右下x座標
d3 ＝　右下y座標

に取り出される。25行でサブルーチンgramadrを呼び出し，左上の座標に対応するG-RAM上のアドレスを得る。それから，27～28行で描くべき長方形の横のドット数－1と縦のドット数－1を求め，これをループカウンタの初期値とする。30行で描画色のパレットコードをd0.wに，32行で水平ライン間のG-RAM上でのアドレスの差をd1.wに入れているのは，アルゴリズム上必須というわけではなく，単に，ループの内側で頻繁に使う定数をレジスタに入れておいて余分なメモリアクセスをしなくても済むようにしているにすぎない。

34行以降がG-RAMに書き込みを行う2重ループだ。36～37行の内側のループで水平方向に線分を引き，それを34～39行の外側のループで必要なライン数分繰り返している。水平方向の線分を引く部分では，ポストインクリメントつきアドレスレジスタ間接アドレッシングが綺麗に決まっている。

リスト4　GFILL.S

```
 1:          .include        gconst.h
 2: *
 3:          .xdef   gfill
 4:          .xref   gramadr
 5: *
 6:          .offset 0
 7: *
 8: X0:      .ds.w   1
 9: Y0:      .ds.w   1
10: X1:      .ds.w   1
11: Y1:      .ds.w   1
12: COL:     .ds.w   1
13: *
14:          .text
15:          .even
16: *
17: gfill:
18: PARPTR = 8
19:          link    a6,#0
20:          movem.l d0-d4/a0-a1,-(sp)
21:
22:          move.l  PARPTR(a6),a1   *a1=パラメータ受け渡し領域
23:          movem.w (a1),d0-d3      *d0-d3に座標を取り出す
24:
25:          bsr     gramadr         *G-RAM上のアドレスを得る
26:
27:          sub.w   d0,d2           *d2=横ドット数-1
28:          sub.w   d1,d3           *d3=縦ドット数-1
29:
30:          move.w  COL(a1),d0      *d0=パレットコード
31:
32:          move.w  #GNBYTE,d1      *d1=ライン間のアドレスの差
33:
34: loop1:   movea.l a0,a1           *a1=左端アドレス
35:          move.w  d2,d4           *d4=横ドット数-1
36: loop2:   move.w  d0,(a1)+        *1ドット点を打つ
37:          dbra    d4,loop2        *d4回繰り返す
38:          adda.w  d1,a0           *すぐ下のラインへ
39:          dbra    d3,loop1        *d3回繰り返す
40:
41: done:    movem.l (sp)+,d0-d4/a0-a1
42:          unlk    a6
43:          rts
44:
45:          .end
```

リスト5　FILLTEST.S

```
 1:          .include        doscall.mac
 2: *
 3:          .xref   gfill
 4: *
 5: CSCREEN          equ     16
 6: DOS_GL3          equ     5
 7: *
 8:          .text
 9:          .even
10: *
11: ent:
12:          lea.l   mysp(pc),sp     *spを初期化する
13:
14:          move.w  #DOS_GL3,-(sp)  *画面を512x512,65536色に
15:          move.w  #CSCREEN,-(sp)  *  初期化
16:          DOS     _CONCTRL        *
17:          addq.l  #4,sp           *
18:          move.w  d0,-(sp)        *現在の画面モードを待避
19:
20:          clr.l   -(sp)           *スーパーバイザモードへ
21:          DOS     _SUPER          *  移行
22:          move.l  d0,(sp)         *現在のsspを待避
23:
24:          pea.l   gpara(pc)       *パラメータ受け渡し領域
25:          bsr     gfill           *boxfill実行
26:          addq.l  #4,sp           *
27:
28:          DOS     _SUPER          *
29:          addq.l  #4,sp           *ユーザーモードへ復帰
30:
31:          DOS     _EXIT           *終了
32: *
33:          .data
34:          .even
35: *
36: gpara:   .dc.w   100,100,200,200,12345
37: *
38:          .stack
39:          .even
40: *
41: mystack:
42:          .ds.l   256
43: mysp:
44:          .end    ent
```

さて，リスト4のgfillにはエラーチェックを一切していないため，不当なパラメータを与えて呼び出すと誤動作したり，バスエラーが発生したりする可能性がある。エラーチェックをして実行速度が遅くなるよりは，呼び出し側でつねに正しいパラメータを渡すよう気をつける，というのもひとつの考えかただが，ここでは真面目にエラーチェックする方向で話を進める。どのような場合に誤動作するかをチェックし，個別に対応を考えてみよう。

1) 座標が実画面の範囲外のとき

これはG-RAMの範囲外の変なメモリへ書き込みを行ってしまうことを意味するから，運がよくてバスエラー，運が悪ければ暴走する危険がある。対応策としては，四隅のどれかひとつでも画面の外だったら描画を一切行わないという消極的な案と，画面外にはみ出す部分を切り取って，画面内に収まる範囲のみを描画する（クリッピングする）という案がある。後者が望ましいのはいうまでもない。

2) 座標の大小関係が狂っているとき

リスト4ではX-BASICでいう，

```
fill(511,511,0,0,15)
```

のような指定がされることを考慮していない。座標をデータレジスタに取り出した時点で，d0≦d2，d1≦d3であるものと思い込み，27〜28行でd2−d0，d3−d1を求めループカウンタの初期値としている。d0＞d2やd1＞d3の場合は減算の結果が負となりループカウンタとしては使いものにならないのに，そのチェックを怠っているのだ。つねに正しい動作を保証するためにはあらかじめ座標の大小関係を調べ，必要ならd0とd2，d1とd3の値を交換しておかなければならない。ちなみに，レジスタ内容の交換にはexgという命令が使える。

リスト6にgfill用のクリッピングサブルーチンを示す。座標はd0〜d3に入れて渡し，クリッピングされた座標がやはりd0〜d3に返る。ただし，指定された長方形が完全に実画面の外にある場合はccrのNビットを立てて戻ることにしてある。クリッピングの処理上，d0≦d2，d1≦d3でないと不都合があるので，どさくさにまぎれ，座標の大小関係の乱れも直している。なお，X-BASICやIOCS同様に座標の範囲を−32768〜32767とした関係で，座標の比較はすべて符号つき16ビット数として行っている。

では，リスト5にクリッピングサブルーチンの呼び出しをつけ加えておく。23行の後ろに，

```
bsr     gfclip
bmi     done
```

の2行を挿入し，また，5行のあたりにでも，

```
.xref   gfclip
```

の1行をつけ加える。修正が済んだら，ふたたび動作試験をしてみよう。いろいろと座標値を変えてうまくクリッピングできているかどうか確かめてほしい。

サブルーチンの高速化

サンプルとはいえせっかく作ったサブルーチンだ。速度と機能の両面で少しパワーアップしてやろう。順序が逆のような気がしないでもないが，まず高速化から考える。アルゴリズムに関してはもともと単純なのでこれ以上最適化しようがない。コーディングのうえでも多少は改善の余地があるにせよ，劇的な効果は望めそうもないように見える。ループの中身で無駄なことをしているようならそれをループの外に追い出すことでかなりの速度向上が望めるのだが，リスト4では，最も内側のループは，

```
loop2:  move.w  d0,(a1)+
```

リスト6 GFCLIP.S

```
 1:         .include        gconst.h
 2: *
 3:         .xdef   gfclip
 4: *
 5: MINMAX  macro   data1,data2
 6:         local   skip
 7:         cmp.w   data1,data2     *data1 > data2ならば
 8:         bge     skip            *
 9:         exg.l   data1,data2     *  data1とdata2を交換する
10: skip:
11:         endm
12: *
13:         .offset 0
14: *
15: MINX:   .ds.w   1
16: MINY:   .ds.w   1
17: MAXX:   .ds.w   1
18: MAXY:   .ds.w   1
19: *
20:         .text
21:         .even
22: *
23: *       (d0.w,d1.w)-(d2.w,d3.w)を
24: *       cliprectで指定された矩形領域でクリッピングする
25: *
26: gfclip:
27:         move.l  a0,-(sp)
28:         lea.l   cliprect(pc),a0
29:
30:         MINMAX  d0,d2           *d0<d2を保証する
31:         MINMAX  d1,d3           *d1<d3を保証する
32:
33:         cmp.w   MAXX(a0),d0     *d0>MAXX ?
34:         bgt     outofscrn       *  そうなら画面外
35:         cmp.w   MAXY(a0),d1     *d1>MAXY ?
36:         bgt     outofscrn       *  そうなら画面外
37:
38:         cmp.w   MINX(a0),d2     *d2<MINX ?
39:         blt     outofscrn       *  そうなら画面外
40:         cmp.w   MINY(a0),d3     *d3<MINY ?
41:         blt     outofscrn       *  そうなら画面外
42:
43:         cmp.w   MINX(a0),d0     *d0<MINX ?
44:         bge     skip1           *
45:         move.w  MINX(a0),d0     *  そうなら修正
46:
47: skip1:  cmp.w   MINY(a0),d1     *d1<MINY ?
48:         bge     skip2           *
49:         move.w  MINY(a0),d1     *  そうなら修正
50:
51: skip2:  cmp.w   MAXX(a0),d2     *d2>MAXX ?
52:         ble     skip3           *
53:         move.w  MAXX(a0),d2     *  そうなら修正
54:
55: skip3:  cmp.w   MAXY(a0),d3     *d3>MAXY ?
56:         ble     skip4           *
57:         move.w  MAXY(a0),d3     *  そうなら修正
58:
59: skip4:  cmp.w   d0,d0           *N=0
60: done:   movea.l (sp)+,a0
61:         rts
62: *
63: outofscrn:
64:         moveq.l #-1,d0          *N=1
65:         bra     done
66: *
67: cliprect:
68:         .dc.w   0               *クリッピング領域
69:         .dc.w   0               *
70:         .dc.w   GNPIXEL-1       *
71:         .dc.w   GNPIXEL-1       *
72:
73:         .end
```

```
        dbra    d4, loop2
```

というこれ以上は簡単にならない形だし，外側のループにも極端な無駄は見られない。唯一，34行のmoveaは工夫次第では省略できそうだ。が，36～37行のループで食う時間に比べれば転送命令1つの実行時間なんか気にならない程度ということで無視してしまおう。

どうも真正直にやっていたのではそれほど速くなりそうもないので，ループを展開するという姑息な手段を検討してみる。話を簡単にするために描く長方形の横幅はつねに512ドットということにし，最も内側のループだけを取り出して考える。いまこのループは，

```
        move.w  #512-1, d4
loop:   move.w  d0, (a0)+
        dbra    d4, loop
```

という形になっているわけだが，これを，

```
        move.w  #512/2-1, d4
loop:   move.w  d0, (a0)+
        move.w  d0, (a0)+
        dbra    d4, loop
```

のように書き直せば，ループ回数，すなわちdbraの実行回数を半分に減らすことができる。さらには，

```
        move.w  #512/4-1, d4
loop:   move.w  d0, (a0)+
        move.w  d0, (a0)+
        move.w  d0, (a0)+
        move.w  d0, (a0)+
        dbra    d4, loop
```

とすればdbraの実行回数は4分の1だ。この話をずーっと突き詰めていくと，

```
        move.w  d0, (a0)+
```

を512個並べてしまえば，dbraを一度も実行しなくて済む計算になる。ループを展開するとはこういうことだ。

当然のようにプログラムはだらだらと長くなる。

```
        move.w  d0, (a0)+
```

は1ワード命令だから，512個並べれば1024バイトだ。これは通常のサブルーチンの大きさと比べれば決して小さな数字ではない。しかし，X68000のメインメモリは最低でも1Mバイト，最近では2Mバイトが当たり前だから，割合としては無視できる範囲だといえる。ループの展開は芸もなければ品もない，テクニックと呼ぶのも恥ずかしい，姑息以外のなにものでもない手段ではあるが，それによってもたらされる速度の向上は大きな魅力だ。あくまでも最後の切り札として，多少は照れたフリをしながら使うようにしたい。

さて，512ドットの水平線を描くだけならこれで話は終わりだが，現実には線分の長さは可変だ。このあたりへの対応も考えておこう。いま，512個の，

```
        move.w  d0, (a0)+
```

とrtsからなるhlineというサブルーチンがあったとしよう。a0がG-RAM内を指しているとすると，hlineの先頭から実行すればd0.wで指定されたパレットコードで512ドットの水平線が描けるわけだ。それでは，hlineの先頭のmoveを1個飛ばして，その直後から実行すればどうなるだろうか。そう，511ドットの線分が描ける。2個飛ばせば510ドットだ。さっきも触れたように，

今月登場した68000の命令

リスト1で使ったext命令はおそらく初登場だと思う。この命令はデータレジスタを強制的に符号拡張する命令だ。符号拡張とは，符号つき数の正負と値を変えずに，より長いビット長のデータに変換することだった。

```
        ext.w   d0
```

は，d0.bを符号拡張して16ビットデータにし，d0.wに結果を残す。2の補数表現での符号拡張は，上位ビットを元のデータの符号ビットで満たすのと等価だから，上の命令は，

```
        tst.b   d0
        bmi     minus
        andi.w  #$00ff, d0
        bra     next
minus:  ori.w   #$ff00, d0
next:   ～
```

を1命令で実行してしまうものと思ってもよいだろう。また，

```
        ext.l   d0
```

はd0.wを32ビットに符号拡張してd0.lに結果を残す。この動作は，

```
        tst.w   d0
        bmi     minus
        andi.l  #$0000ffff, d0
        bra     next
minus:  ori.l   #$ffff0000, d0
next:   ～
```

に等しい。

つづいて，リスト4などのmovemについても触れておこう。movemは，

```
        movem.l d0-d7/a0-a5, -(sp)
            ⋮
        movem.l (sp)+, d0-d7/a0-a5
```

のようにしてレジスタをまとめてスタックに待避・復帰するのに使う命令としてもうお馴染みだが，リスト4の23行では，

```
        movem.w(a1), d0-d3
```

と，いつもとは少し違う使い方をしている。

まず，ソースオペランドは見慣れた“(sp)+”ではなく“(a1)”で，しかもオペランドサイズはワードだ。この命令の意味するところは“a1でポイントされるメモリ以降の4ワード，(a0)，2(a0)，4(a0)，6(a0)を順にd0～d3に転送する”だ。“(a1)+”のようにポストインクリメントを指定しているわけではないので，命令実行後もa1の値は変化しない。逆に，

```
        movem.w d0-d3, (a1)
```

なら，a1以降の連続するメモリにd0.w～d3.wが転送される。フラグの変化を無視すれば，これは，

```
        move.w  d0, (a1)
        move.w  d1, 2(a1)
        move.w  d2, 4(a1)
        move.w  d3, 6(a1)
```

と同じ働きをする（moveではccrが変化するが，movemではccrは変わらない）。

ここで，もう一度

```
        movem.w (a1), d0-d3
```

に話を戻す。これまでの話からすると，この命令は，

```
        move.w  (a1), d0
        move.w  2(a1), d1
        move.w  4(a1), d2
        move.w  6(a1), d3
```

と同じ動作をするように思えるだろう。が，実はちょっと違う。正しくはこのあとに，

```
        ext.l   d0
        ext.l   d1
        ext.l   d2
        ext.l   d3
```

をつけ加えなければならない。結果は無条件にロングワードに符号拡張され，データレジスタの全32ビットが影響を受けるのだ。これは案外盲点なので気をつけるようにしたい。

あと，リスト6などでは符号つき数の比較時に使うbraのバリエーションがちらほら顔を出している。『アセンブラマニュアル』などで働きをチェックしておいてほしい。

move.w　d0, (a0)＋

は1ワードの命令だから,

hline＋ (512－描きたいドット数)×2

のアドレスから実行すれば，希望どおりの長さの線分を描くことができる。サブルーチン末のrtsが置かれたアドレス(仮にhline0とすると)を基点と考えるとこの式はもう少し簡単に表せ,

hline0－描きたいドット数×2

となる。

リスト7がループを展開したgfillの新しい版だ。リストの短さは意外かもしれない。展開したループの中身はどこにいってしまったかというと，45行に凝縮されている。この.dcbは同じ値の定数が並んだブロックを用意する疑似命令で，45行では$30C0_H$というワードデータをGNPIXEL個だけ用意している。ここで，$30C0_H$は,

move.w　d0, (a0)＋

のマシンコードであり，GNPIXELはGCONST.H内で定義された実画面の横ドット数を表す記号定数だ。

実はgfillは,

move.w　d0, (a0)＋

で1ドットずつ点を打つ代わりに,

move.l　d0, (a0)＋

で2ドットまとめて描くというアイデアによってさらに高速化できる。ただ，横ドット数がつねに偶数とは限らないから，奇数の場合は最初か最後の1ドットを特別扱いする必要が出てくる。この点に注意しプログラムにしてみたらリスト8のようになった。リスト7からの変更部分に注目して読んでみてほしい。

描画モードを加えよう

速度の追求はこのくらいにして，こんどは機能の拡張に目を向ける。描画モードが使えるようにするのは非常に簡単だ。moveでG-RAMに書き込んでいる部分をorにするだけでORモードになるし，eorにすればXORモードになる。パラメータで描画モードを指定し処理を振り分けるよりは直接リスト8の部

リスト7　GFILL2.S

```
 1:          .include        gconst.h
 2: *
 3:          .xdef   gfill
 4:          .xref   gramadr
 5:          .xref   gfclip
 6: *
 7:          .offset 0
 8: *
 9: X0:      .ds.w   1
10: Y0:      .ds.w   1
11: X1:      .ds.w   1
12: Y1:      .ds.w   1
13: COL:     .ds.w   1
14: *
15:          .text
16:          .even
17: *
18: gfill:
19: PARPTR = 8
20:          link    a6,#0
21:          movem.l d0-d3/a0-a1,-(sp)
22:
23:          move.l  PARPTR(a6),a1   *a1=パラメータ受け渡し領域
24:          movem.w (a1),d0-d3      *d0-d3に座標を取り出す
25:
26:          bsr     gfclip          *クリッピングする
27:          bmi     done            *N=1なら描画の必要なし
28:
29:          bsr     gramadr         *左上のG-RAM上のアドレスを得る
30:
31:          sub.w   d0,d2           *d2=横ピクセル数-1
32:          sub.w   d1,d3           *d3=縦ピクセル数-1
33:
34:          move.w  COL(a1),d0      *d0=パレットコード
35:
36:          lea.l   next(pc),a1     *本文参照
37:          addq.w  #1,d2           *
38:          add.w   d2,d2           *
39:          suba.w  d2,a1           *
40:
41:          move.w  #GNBYTE,d1      *d1=ライン間のアドレスの差
42:          sub.w   d2,d1           *
43:
44: loop:    jmp     (a1)            *1ライン描画
45:          .dcb.w  GNPIXEL,$30c0   * move.w d0,(a0)+
46: next:    adda.w  d1,a0           *すぐ下のラインへ
47:          dbra    d3,loop         *d3回繰り返す
48:
49: done:    movem.l (sp)+,d0-d3/a0-a1
50:          unlk    a6
51:          rts
52:
53:          .end
```

リスト8　GFILL3.S

```
 1:          .include        gconst.h
 2: *
 3:          .xdef   gfill
 4:          .xref   gramadr
 5:          .xref   gfclip
 6: *
 7:          .offset 0
 8: *
 9: X0:      .ds.w   1
10: Y0:      .ds.w   1
11: X1:      .ds.w   1
12: Y1:      .ds.w   1
13: COL:     .ds.w   1
14: *
15:          .text
16:          .even
17: *
18: gfill:
19: PARPTR = 8
20:          link    a6,#0
21:          movem.l d0-d3/a0-a2,-(sp)
22:
23:          move.l  PARPTR(a6),a1   *a1=パラメータ受け渡し領域
24:          movem.w (a1),d0-d3      *d0-d3に座標を取り出す
25:
26:          bsr     gfclip          *クリッピングする
27:          bmi     done            *N=1なら描画の必要なし
28:
29:          bsr     gramadr         *左上のG-RAM上のアドレスを得る
30:
31:          sub.w   d0,d2           *d2=横ピクセル数-1
32:          sub.w   d1,d3           *d3=縦ピクセル数-1
33:
34:          move.w  COL(a1),d0      *d0=パレットコード
35:          swap.w  d0              *
36:          move.w  COL(a1),d0      *
37:
38:          lea.l   next(pc),a2     *a2=戻りアドレス
39:          addq.w  #1,d2           *
40:          bclr.l  #0,d2           *横ドット数は奇数か？
41:          beq     skip            *
42:          lea.l   odd(pc),a2      *奇数ドットのとき
43:
44: skip:    lea.l   hline0(pc),a1   *
45:          suba.w  d2,a1           *
46:
47:          move.w  #GNBYTE,d1      *d1=ライン間のアドレスの差
48:          add.w   d2,d2           *
49:          sub.w   d2,d1           *
50:
51: loop:    jmp     (a1)            *1ライン描画
52: odd:     move.w  d0,(a0)         *奇数ドットの場合
53: next:    adda.w  d1,a0           *すぐ下のラインへ
54:          dbra    d3,loop         *d3回繰り返す
55:
56: done:    movem.l (sp)+,d0-d3/a0-a2
57:          unlk    a6
58:          rts
59: *
60: hline:
61:          .dcb.w  GNPIXEL/2,$20c0 *move.l d0,(a0)+
62: hline0:  jmp     (a2)
63:
64:          .end
```

表1　各種描画モードへの対応（変更部分）

モード	行	ラベル	命令	コメント
ORモード	52:	odd:	or.w　d0, (a0)	＊奇数ドットの場合
	61:		.dcb.w　GNPIXEL/2, $8198	＊or.l d0, (a0)＋
XORモード	52:	odd:	eor.w　d0, (a0)	＊奇数ドットの場合
	61:		.dcb.w　GNPIXEL/2, $b198	＊eor.l d0, (a0)＋
ANDモード	52:	odd:	and.w　d0, (a0)	＊奇数ドットの場合
	61:		.dcb.w　GNPIXEL/2, $c198	＊and.l d0, (a0)＋

分部分を修正して別のサブルーチンにしてしまったほうが楽だろう。というわけで，OR, XOR, AND各モードへの変更箇所を表1にまとめておく。表には特に示していないが，サブルーチン名もgfill_orとかなんとか，それらしい名前に変更しておくとよい。そのときは同時に3行の外部定義名も変えるのを忘れないように。

これらgfillのバリエーションの動作試験にはリスト5を改造したものを使ってもよいが，先月のLINE.Sを流用してそれぞれ独立したプログラムにしておけば便利だろう。先月のリスト4の7行を，

```
PARCNT  equ     5
```

に，22～25行のIOCSコール呼び出し部分を，

```
clr.l       -(sp)
DOS         _SUPER
move.l      d0,(sp)
pea.l       giospar(pc)
bsr         gfill_or            *などなど
addq.l      #4,sp
DOS         _SUPER
addq.l      #4,sp
```

のように修正し，さらに5行の直後あたりにでも，

```
.xref       gfill_or            *などなど
```

と外部参照定義を加える。あとは個別にアセンブルしたうえで必要なサブルーチンとリンクすれば出来上がりだ。

おまけはタイリングペイント

もうひとつだけgfillのバリエーションを作って今月は終わりにしたい。いわゆるタイリングができるようにしてみる。タイリングは異なる色を交互に並べることで解像度をある程度犠牲にして疑似的に多色表示を行う手段だ。65536色が同時に出せるX68000では無用といえばそうなのだが，16色モードや256色モードではまだまだ使い道はあると思う。

あまり複雑にはしたくないので，2色の市松模様専用に限定して考えよう。要は2色の点を交互に並べていけばよいわけだ。運よく，リスト8のgfillでは2ドットをまとめて書き込むようになっているので，d0.lの上位ワードと下位ワードに2色のパレットコードを入れておけば簡単にタイリングが実現できることになる。あとは実際のコードを見てもらったほうがよくわかるだろう。で，リスト9だ。

基本的にはリスト8をベースに最小限の修正を加えた形だ。ここでも変更部分に注目して読んでみてもらいたい。データレジスタの上位ワードと下位ワードを交換するswap命令の使い方がポイントだ。各swap命令の意味がつかめれば，プログラム全体の動作も見えてくるだろう。

＊

えーと，『猫ふんじゃった』って曲があるじゃない？　ピアノ教室に嫌々通っている子供でもなぜかあの曲だけは喜んで弾くというあれだ。あの曲は単純な指遣いながら黒鍵は多用するし，手を交差させたりもなんかして，ピアノを弾いている気分を味わえるというのが人気の秘密だって昔なにかで読んだ記憶がある。いま，なんとなく思い出した。

さて，次回もグラフィック，今回の話のつづきになる模様だ。それでは，また来月。

リスト9　TILEFILL.S

```
 1:          .include        gconst.h
 2: *
 3:          .xdef   gtilefill
 4:          .xref   gramadr
 5:          .xref   gfclip
 6: *
 7:          .offset 0
 8: *
 9: X0:      .ds.w   1
10: Y0:      .ds.w   1
11: X1:      .ds.w   1
12: Y1:      .ds.w   1
13: COL1:    .ds.w   1
14: COL2:    .ds.w   1
15: *
16:          .text
17:          .even
18: *
19: gtilefill:
20: PARPTR = 8
21:          link    a6,#0
22:          movem.l d0-d3/a0-a2,-(sp)
23:
24:          move.l  PARPTR(a6),a1   *a1=パラメータ受け渡し領域
25:          movem.w (a1),d0-d3      *d0-d3に座標を取り出す
26:
27:          bsr     gfclip          *クリッピングする
28:          bmi     done            *N=1なら描画の必要なし
29:
30:          bsr     gramadr         *左上のG-RAM上のアドレスを得る
31:
32:          sub.w   d0,d2           *d2=横ピクセル数-1
33:          sub.w   d1,d3           *d3=縦ピクセル数-1
34:
35:          eor.w   d0,d1           *
36:          move.l  COL1(a1),d0     *d0=パレットコード
37:          btst.l  #0,d1           *
38:          beq     skip0           *
39:          swap.w  d0              *
40:
41: skip0:   lea.l   next(pc),a2     *a2=戻りアドレス
42:          addq.w  #1,d2           *
43:          bclr.l  #0,d2           *横ドット数は奇数か？
44:          beq     skip            *
45:          lea.l   odd(pc),a2      *奇数ドットのとき
46:
47: skip:    lea.l   hline0(pc),a1   *
48:          suba.w  d2,a1           *
49:
50:          move.w  #GNBYTE,d1      *d1=ライン間のアドレスの差
51:          add.w   d2,d2           *
52:          sub.w   d2,d1           *
53:
54: loop:    jmp     (a1)            *1ライン描画
55: odd:     swap.w  d0              *奇数ドットの場合
56:          move.w  d0,(a0)         *
57:          swap.w  d0              *
58: next:    adda.w  d1,a0           *すぐ下のラインへ
59:          swap.w  d0              *
60:          dbra    d3,loop         *d3回繰り返す
61:
62: done:    movem.l (sp)+,d0-d3/a0-a2
63:          unlk    a6
64:          rts
65: *
66: hline:
67:          .dcb.w  GNPIXEL/2,$20c0 *move.l d0,(a0)+
68: hline0:  jmp     (a2)
69:
70:          .end
```

X68000用

ハンディイメージスキャナアダプタの製作

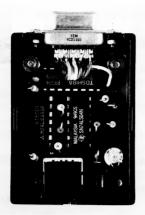
Hayashi Yohzow
林 曜三

ポータブルワープロのスキャナって，結構互換性があるのです。ここでは安価で手軽なワープロ用のハンディイメージスキャナをX68000に接続するためのアダプタを製作してみましょう。当然のことながら，これをドライブする専用の読み取りソフトもあわせて発表します。

ハンディスキャナがほしい

突然ですが，世間ではイメージスキャナが流行ってますね。FAX，ハンディコピー，ペン型翻訳機なんてのもCCD[1]の発達で小型化されたスキャナ(ラインセンサ)の応用製品です。そしてなんといっても，ワープロ，パソコンといった情報機器は，そのほとんどがオプションとしてイメージスキャナを持つようになりました。某386SXマシンには標準でインタフェイスがついてますね。でも，その用途は，大半は「文章にちょっとした写真やカットを入れる」というワープロの機能拡張的なものですが。

さて，X68000に目を移しましょう。ご存じのとおり，X68000にはCZ-8NS1というA4をフルカラーで読めるイメージスキャナがあります。が，高い。白黒のスキャナはRS-232Cタイプのものが使えますが，相対的に高価で，ドライバソフトもついてくるわけではありません。よって，スキャナを持っている人は当然少数です(よね？)。

しかも，いまの時点でイメージを張り付けられるワープロはなく（無理すればできますが)，当然その用途に使う人もいませんからさらにスキャナを買う人は少なくなります。

しかし，しかしです。X68000は，映像関係に標準で1Mバイト以上という某一太郎マシン＋フレームバッファ[2]をも凌駕するRAMを備えた，映像をウリにしたパソコンです。安価なスキャナがないため，どれだけの人がその機能を生かせなくて泣いていることか（おおげさ)。某一太郎マシンには専用の，26万色をわざわざ8色に落とすという超無駄オーバースペックハンディカラースキャナがあるというのに，なぜX68000には白黒のスキャナさえないのだあぁ。シャープの人に聞いたら，X68000用にハンディスキャナを出す予定はないとのこと。あっさり夢も希望も打ち砕かれたところで，本題に入りましょう。

投げ売りスキャナに愛の手を

というわけで，ないものは作ってしまえというフロンティアスピリット（ああ，これを忘れたらパソコンはマイコンじゃなくなる）にのっとってハンディスキャナをなんとかつなげてみました。というのは建て前で，ほんとは今年初め，秋葉原でワープロ用のスキャナが4,800円で投げ売り（にしては高いかなあ）していたので衝動買いしてつなげたのです。インタフェイスを超ケチッてアダプタ形式にしたので，完全無欠とはいきませんが，やはり腐っても鯛ですね(ひどいいい方)。X68000ならではの美しい画像は他機種の追随を許しません。え？ スキャナの画像に機種で差がつくものかって？

まあコラムをご覧ください。

とにかく，IC3個という小さなセットですので，本誌のハードウェア工作入門のちょっとした応用編と考えて，ぜひ作ってみてください。

あなたもわたしもOEM

さて，気になる対応スキャナですが，アルプス電気のOEM[3]で（外から見ただけ

読み取り画像の情報量について

スキャナの読み取りプログラムを設計する場合，問題になるのが読み取るドット数です。一般的なスキャナの解像度は1000画素前後(CCDの解像度はさらに上)ですが，世間のワープロやパソコンは画面の解像度がそんなにありません。実際には512ドットまたは640ドットしか読んでいないのです。

私は今回手に入れたワープロ用スキャナの説明書を見て思わず笑ってしまいました。解像度を400dpiにすると読み取り幅はたった32㎜になってしまうのです。パソコンでも水平640ドット程度じゃ今主流の105㎜を読もうとすると解像度を150dpi以下にしなければなりません。宝の持ち腐れとは正にこのことです。

X68000ではそんなことはありません。実画面サイズ1024×1024ピクセルの水平型VRAMという，まるでスキャナの読み取り用に設計されたのではと錯覚するテキスト画面が4枚もあります。これを使えば最大1024×4096ドットという広範囲をCRTで確認しながら読み込めるのです。

今回はさすがにそこまではやりませんでしたが，CRTのアスペクト比（2:3）に合うように最大1024×1536ドットを読み取り，縮小するときに多値化しています。つまり，最高6倍のオーバーサンプリングを行い，見かけの解像度を上げているわけです。

ここでちょっと読み取り画像の情報量について調べてみましょう。n色の画像1画素当たりの情報量は$\log_2 n$（ビット）で表されますから，標準的な，640×400ドット白黒の画像では，情報量は最大で，$640\times400\times\log_2 2=250$Kビットとなります。

では，今回のプログラムで，原画像サイズ1024×1536ドットを768×512ドット，5階調に変換した場合はどうでしょうか。

$768\times512\times\log_2 5\fallingdotseq 891$Kビット

となり，実に3.5倍以上の情報量を持つことがわかります。近くにハンディスキャナのつながっているワープロなりパソコンのある方は，ぜひ同じ原稿で見比べてみてください。スキャナの持つ真の実力に驚かれることでしょう。

石が3つの簡単な基板だ

じゃわかりませんが)，コネクタが8ピンミニDIN(X68000のキーボードなどのコネクタと外見が同一)のものはまずつながるのではないかと思われます。この規格はモノクロのスキャナではほぼ業界標準ともいえるほど普及しています。文書のフォーマットなどほとんど互換性のないワープロ界ですが，ことスキャナのこととなると，ほとんどの機種に互換性があります。

私が購入したのはキヤノンのキヤノワードαシリーズ用のCW-IM02というものですが，これはひと昔前，まだ64mm幅のものが主流だった頃ポピュラーだったモデルで，いまでもパソコン用でHAL研，NEOSなどのモデルに採用されています。スペックとしては，64mm幅，400dpi[4]，3種類の64階調ディザリング機能を持っています。また同じインタフェイスで，ディザリング機能のない廉価版や，最近主流になってきた105mm幅のもの，カラー(3度読み)のもの，またビデオ画像を取り込むビデオスキャナなども各社から発売されているようです。

とにかく規格

さて，スキャナが手に入ったところでインタフェイスの設計を始めるわけですが，その前にスキャナ側の規格がわからないことにはお話になりません。幸い，友人がNEOS製のPC-9801用インタフェイスボードを持っていたのでそれで調べることにしました。

最近のボードはカスタムICが載っている場合が多く見受けられますが，これはすべてディスクリート(離散)部品でできており，簡単に動作が理解できました(結構タコな部分もありましたが)。

図1に，各端子の機能を示します。では詳しく説明しましょう。1番は+12Vの電源です。消費電流は400mAくらい(ほとんどが光源用)流れますがものによっては(カラーとか自動走査とかのものは)1A近く流れるものもありますので注意が必要です。2番と8番はグランド，0Vです。

3番は主走査(横方向の，CCDによるスキャン)の始まりを立ち上がり(ローからハイになる)で知らせます。

4番は8ドットごとにローになる信号で，メモリに直結する場合，これでアドレスをカウントします(親切だなあ)。

5番はドットクロックです。立ち上がりのときに有効な画像データが6番にあります。周期は実測値で約1.3μ秒でした(770kbps!)。

7番は人が手で行う縦方向のスキャンの同期信号で，立ち下がりで新しいラインに入ったことを知らせます。これらの様子を図2に示します。

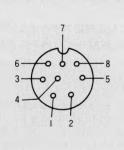
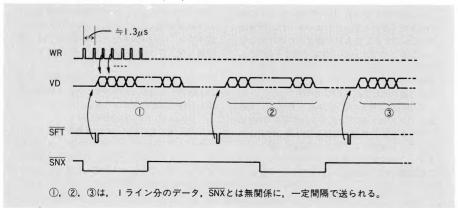
図1　スキャナ側から見た信号

ピン番号	信号名	I/O	備　考
1	+12V	IN	電源
2	GND	IN	
3	$\overline{\mathrm{SFT}}$	OUT	主走査スタートタイミング
4	$\overline{\mathrm{AU}}$	OUT	アドレスアップ
5	WR	OUT	データ取り込みタイミング
6	VD	OUT	ビデオデータ(H黒/L白)
7	$\overline{\mathrm{SNX}}$	OUT	副走査同期
8	GND	IN	

図2　スキャナ信号タイミングチャート

①，②，③は，1ライン分のデータ，$\overline{\mathrm{SNX}}$とは無関係に，一定間隔で送られる。

誤算その1

以上を見てもらえればわかるとおり，データは完全に一方的な同期による垂れ流しです。しかも転送速度も決してのんびりした速度ではありません。PC-9801用のインタフェイスでは，8MHzの8086を想定して，16ビットパラレルに変換して読み込んでいましたが，諸々の処理を含めるとそれでも余裕とはいえないようでした。そうなると，X68000でも専用のインタフェイスボードを作らなきゃならないような気がしてきて，実際試作してみました。ジョイスティックポートの利用も最初考えたのですが「速度が間に合いっこない」と思い込んでしまっていたのです。

ところがある日，ふとジョイスティックポートを利用したインタフェイスを，だめもとで作ってみる気になったのです。いや，人間諦めが早いと損しますね。動くじゃないですか。シシリアンの設計者に感謝！　というわけでボードはまったくの無駄になってしまいましたとさ。

アダプタの概要

X68000のジョイスティックポートは正しくATARI規格です。回路図などの文献を見ればわかりますが，連続して取り込めるビットは4ビットしかありません。今回のようにパラレル(並列)ポートとして使う場合，これがネックになってきます。X1のように8ビットの汎用ポートなら嬉しかった(?)のですが，ない袖は振れませんので4ビットで頑張るしかありません。

あとは，1.3×4=5.2μ秒で4ビットという速度にソフトウェアが追いつくかどうか，です。これが可能であるとして，アダ

1) CCD
電荷結合素子のこと。電子をバケツリレーのように運ぶシフトレジスタのことなのですが，光電素子と組み合わされて撮像素子に応用されることが多いので，一般にCCDといえば撮像素子を指すようになりました。
2) フレームバッファ
要するに大容量のG-RAM。640×400ドット，1677万色(24ビット/ピクセル)というのが標準的。ACRTCを積み，整数倍の拡大やスムーススクロールをハードでサポートするものが多い。
3) OEM
相手先ブランドによる販売。最近大流行(だそうだ)。
4) dpi
ドット/インチ。1インチ(25.4mm)の中に何ドットあるか，という分解能を表す単位。400dpi≒16ドット/mm

プタは，スキャナからシリアルに送られてくる画像データを４個ずつまとめて渡してやるという作業をすればいいのです。

しかし，データは垂れ流しですから，このままではX68000は次のデータがいつきているのかわかりません。そこでここからは次の４ビット，と知らせる信号（同期信号）が必要です。さらに図１に示したとおり，凶悪なことに“次の１ライン分のデータ＝次のライン，とは限らない”ので，次のラインを知らせる副走査信号も必要です。この２つの信号は余ったトリガボタン用ビットで渡します。

実際の回路

以上のことをまとめたのが図３の回路です。簡単でしょう。以前のボードでIC14個の規模だったのがたったの３個！（単純に比べられないが）そう，私のハード製作の信条はいかに安くして最大のパフォーマンスを得るか，です（単に貧乏なだけという話もある）。

わかる人は一瞬でわかると思いますが，初心者の方のために一応説明します。

まず箱が３個ありますね。それからなにやら三角形に丸のついたのがいっぱいあります。これらが実はICのひとまとまりの機能を表しています。

いちばん上の箱の上にHC595と書いてありますが，これはひとつのICで，HC595というのはICの型番です。このICは８ビットのシフトレジスタに３ステート出力のラッチがついたものです。要するに，シリアル→パラレル変換をする心臓部です。８ビットというのは無駄なようですがほかに適当な石がありませんので。その代わり，８ビットのポートにも少し手を加えるだけでつながります。箱の内部にいろいろ記号が書いてありますが，それが各端子の機能です。

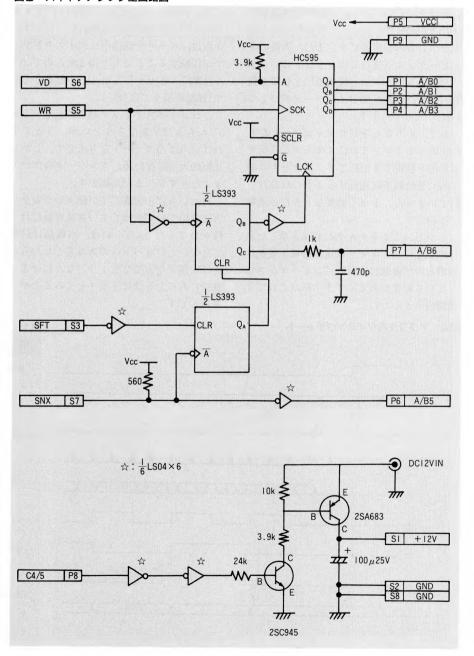
詳しくは規格表を見てもらうことにして，簡単に説明しますと，Aというのがシリアル信号の入力です。ここにスキャナからの画像信号を入れてやります。SCKはシフトクロックの略で，ここを立ち上げると内部でデータがひとつシフトして，A端子からデータが取り込まれます。SCKの前に＞がついていますが，これは信号のエッジ（変化点）での動作を表しています。これをスキャナの読み込みタイミング信号につないでやります。LCKはラッチクロックです。これを立ち上げると，初めてシフトレジスタの内容が出力であるＱＡ～ＱＢに現れます。そして次にLCKを入力するまで変わりません（これをラッチ＝掛けがね動作という）。

それでは次に，その下の1/2LS393と書かれた２つの箱を見てみましょう。これらは２つで１個のICに入っています。そのことを明確にするために1/2と先頭につけてあるのです。LS393が型番です。さて，この箱は４ビット（16進）のカウンタです。動作は，CLRでカウンタのクリア，Aは立ち下がりでカウントアップをします。Aの前の＞はいいとしてそのさらに前に丸がついてますが，これは負論理（ローで有効）を表します。つまり，立ち下がりが有効なのですね。QA～QDは出力です。

上のLS393は分周器の役割を果たしています。スキャナの読み込みクロックを数えて，４クロックごとにHC595のラッチを動作させて出力データを更新し，新しいデータであることをホスト（X68000）に知らせています。

まあ，これはいいのですが下のほうのLS393は変なことに使っています。IC数節約のため強引にRSフリップフロップ[5]として使っているのです。つまり，副走査同期

5) RSフリップフロップ
フリップフロップとはシーソーのこと。入力によってこのシーソーはどちらかに倒れ，シーソーは次になにかが入力されるまでその状態を保持する。概念的にはそんな感じです。

図３ スキャナアダプタ全回路図

信号が立ち下がってから主走査スタート信号がくるまでのあいだ，カウンタをクリア(カウントアップさせないように)しているわけです。

上のLS393のQCに抵抗とコンデンサがつながっていますが，これはHC595のラッチ動作がQCの変化よりも幾分遅れるため，それにタイミングをあわせるための積分遅延器です(図4参照)。ただし，これは入力ポートが15kΩでプルアップされているという前提で設計しています。ですからX68000以外の機種では変更する必要があるかもしれません（バッファが足りなかったんです)。

最後のICはインバータ（論理反転器)です。ところどころにある三角形がそうです。これはその名のとおり，正論理を負論理に，またはその反対にします。負論理のほうに丸をつけるのが常識です。また，回路のインピーダンス（抵抗分のようなもの）を下げたりするバッファ（緩衝器）としても使われます。合計6個のインバータがひとつのICに入っています。

さて，いちばん下にゴチャゴチャしたものがありますが，これはスキャナに供給する電源を制御する回路です。スキャナは基本的に長時間連続運転するようには設計されていません。よって必要なときだけ通電するのです。

しかし，ここをチャチに作るとひどい目にあいます。スキャナは電源の電圧変動に敏感なので電源は十分低インピーダンスにしてやる必要があるのです（私はこれで2週間悩みました)。

12V電源のこと

スキャナを駆動するのにDC(直流)12Vの電源が必要です。しかも場合によっては1A程度の容量が必要になるかもしれません。そこで，この電源をどうやって確保するかが問題です。パソコン用のスキャナで，初めからACアダプタがついてきた場合を除くと，以下のような方法が考えられます。

1)　X68000本体から取る
2)　1)と同じだがDC–DCコンバータを使う
3)　市販のACアダプタを使う
4)　電源を作る

1)はたとえばイメージ入力端子などから出ている＋12Vの電源を使おうというものです。X68000内部では，12Vはディスクドライブと空冷ファンに使われています。今回は，スキャナ動作中にディスクドライブが作動することはあり得ませんので，0.4A程度のスキャナならなんとかなりそうです(保証の限りではない)。

2)はジョイスティック端子にきている＋5Vから＋12Vを作ろうという，うまくすればいちばんスマートな方法です。しかし比較的大容量なため，コンバータ自体が大きくなりすぎ，また高価です。

3)はいちばん安直ですが確実かつ安全な方法なのでおすすめです（電流容量だけは確かめてください)，4)は，市販品は信用できないというパワーのある方でしたらいちばん確実な方法でしょう。なんにせよ，自分にあった方法にトライしてみるのがいちばんです。

部品集めの旅

方針が決まったところで部品を集めましょう。部品表を見てください。

まず小物から。抵抗は1/8W型のもので十分です。カラーコード[6]を覚えておくと便利ですよ。あと回路図にありませんが各ICには電源にパスコンと呼ばれる誤動作防止用のコンデンサをつける必要があります。容量は0.1μF程度なら適当で構いませんが，高周波特性のよい，セラミックコンデンサなどを選ぶ必要があります。間違っても電解コンデンサは使っちゃいけません。抵抗やコンデンサの精度についてはまったく気にする必要はありません。

では次に半導体を見てみましょう。まずトランジスタです。2SC945は非常にポピュラーな石ですのでどこへ行っても50円することはないでしょう。また，相当品として2SC1815などがあります。2SA683も，最大コレクタ電流（Ic）が－1A程度取れればほかの石でも構いません。

親切な店ですと，トランジスタ1種類ごとに定格を書いてあるところがあります。規格表で調べても石の値段はわかりませんので（雑誌の広告を見るという手もあるが)，実際に店に行って，手持ちのスキャナの定格とを比較して，余裕のあるものを探すとよいでしょう。おっと，あまり大きいと，ケースに入らないかもしれませんよ。

さていよいよICです。ICの型番の頭に74がついていますが，これはTTLIC[7]の74シリーズ（または相当品）であることを表

図4　アダプタ内タイミングチャート

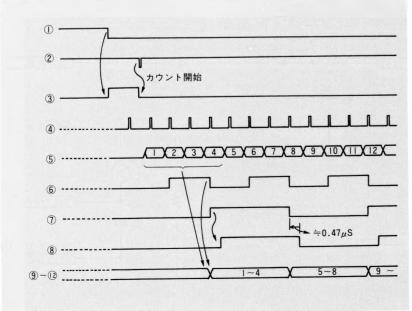



表1　部品表

部品名	[illegible]	数量	備　考
抵抗	560Ω(緑青茶)	1	
	1kΩ(茶黒赤)	1	
	3.9kΩ(橙白赤)	2	
	10kΩ(茶黒橙)	1	
	24kΩ(赤黄橙)	1	
コンデンサ	470pF(471)	1	セラミックコンデンサ
	0.1μF	3	セラミックコンデンサ
	100μF	1	電解(耐圧25V)
IC	74LS04	1	
	74LS393	1	
	74HC595	1	
トランジスタ	2SA683	1	または相当品
	2SC945	1	または相当品
コネクタ	D-SUB(9ピン)メス	1	
	ミニDIN(8ピン)	1	
	電源用コネクタ	1	
万能基板		1	40×60mm以上
ケース		1	SW-55(タカチ電機)等
その他	線材,ビスナット等	適宜	

します。LSはローパワーショットキーバリアという回路形式を表しています。LSの2個のTTLはすぐ手に入るでしょう。問題は，HC595です。HCはハイスピードCMOS[8]の略で，実はTTLの皮をかぶったCMOSなのです。CMOSの弱点であるスピードをLSTTL相当にまで改善し，超低消費電力を実現したシリーズです。

ここでなぜわざわざHCシリーズを使ったかというと，実は595のLSTTL版は，ひとつのメーカーからしか出ていないのです手に入れにくいからです。ただそれだけです。でもHCは入力がCMOSインタフェイスなので，本来はTTLと混在させるのは好ましくありません（入力をプルアップする必要がある→余計な電気を食う）。しかし私のシステムでは一応正常に動くのでそのままにしてあります。お金に余裕のある方はほかの2個のTTLもHCに統一してもよいでしょう。

初心者の方は，必ずICソケットを使いましょう。マルチコンタクトの高級品になると値段が平気でIC自体の何倍もしますが，まあ慣れるまでは世話になったほうがよいと思います。

次にコネクタ類です。絶対必要なのはミニDIN(8ピン)のジャックと，D-SUBコネクタ(9ピン)のメスです。あとは電源のジャックを，手持ちのACアダプタのプラまたは好みにあわせて購入しましょう。

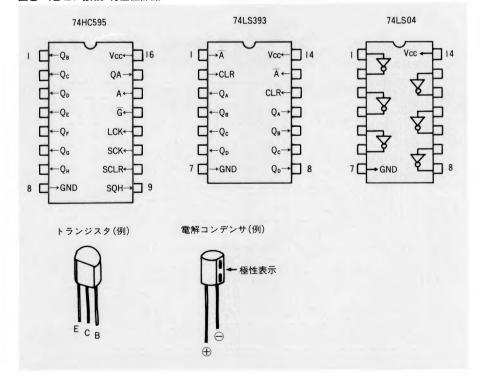
次は基板です。ユニバーサル基板という穴が一面に開けてある基板を使います。もちろん知識のある方ならプリント基板を作ってもよいでしょう（大量生産できる!?）。大きさは，手のひらくらいのものを買ってきて，あとで切断するのがよいと思います。

最後に重要なのがケースです。最近では小さなケースでも沢山バリエーションがありますので，センスの見せどころです。また，ケースに基板を固定するビスとナット，スペーサーなども適当なものを選んでください。

部品が揃いましたら，あとは線材（ラッピングワイヤー[9]が適当）なり半田なりの消耗品も確認しましょう。家に帰ってから「切れてるー」なんてことのないように。

工具について

製作に最低必要な工具は，本誌のハードウェア工作入門で紹介されていますのでそちらを参照してください。ただ，今回のように外付けのセットではケースを加工することになります。耐久性を無視するのなら裸でもいいですけどね。プラスティックのケースなら半田ごてで強引に穴を開けるという無謀な方法も考えられますが，やはり専用の工具を揃えたほうが当然綺麗にいきます。

まず穴開け。ハンドドリルで十分です。ドリルチップは3.2㎜前後のものがあればとりあえず大丈夫ですが，この際セットを揃えておく手もあります。次に，穴をぐりぐり広げるリーマーも必需品です。取っ手のついたものが便利です。また，アルミシャーシを加工するのでしたらハンドニブラー（かじる!?道具）もほしくなります。

半田付けの達人

部品と工具さえ揃えばもう半分完成したも同然です。嬉々として部品配置を考えます（はたから見たら変態だろうなあ）。部品配置と配線の一例を図6に示します。私

図5 ICピン接続/有極性部品

6）カラーコード

抵抗や一部のコンデンサなどの値を示すのに使われる，数字を色で表すコードです。色と数字の対応は，

黒	…0	緑	…5
茶	…1	青	…6
赤	…2	紫	…7
橙	…3	灰	…8
黄	…4	白	…9

いろいろな語呂合わせがありますが，色の順番は，「色相の増加する方向で，最初は明度が上がり，最後に飽和度が上がる」と覚えるとよいでしょう。抵抗の場合，値表示は4～5本の色帯からなり，普通のカーボン抵抗の場合，

第1色帯 …第1数字
第2色帯 …第2数字
第3色帯 …乗数＝10の第3数字乗
第4色帯 …許容差（無色，銀，金とグレードが上がる）

となっています。

たとえば「橙白赤金」となっていれば，抵抗値は，39×10の2乗，つまり3.9kΩです。最後の金色は誤差の許容値で，±5％を保証するということです。

ちなみにコンデンサの値ですが，数字が直接プリントしてある場合が多いですが，いろいろな表し方があるので混乱しないようにしてください。

- 0.1μFなどとそのまま書いてある。
- 4n7など，nFの位置に"n"が入っている。
- 3桁の数字…抵抗の色帯に同じで仮数2桁に乗数のかたち。pF単位。
- 2桁の数字…pF単位そのまま。

たとえば4n7なら4.7nF＝0.0047μF，221なら220pFです。

7）TTL

トランジスタ・トランジスタ・ロジックの略。つまりはトランジスタ（ここでは狭義のバイポーラトランジスタ）を集積しましたよという意味。もっとも広く使われる論理ICです。

8）CMOS

相補型・金属－酸化膜－半導体の略。シーモスと読む。P型とN型両方のMOS電界効果トランジスタを組み合わせて論理回路を構成しています。非常に消費電力が少ないのが特徴。

9）ラッピングワイヤー

配線方法のひとつにラッピングという方法があります。長ーい足に細いワイヤーを専用の治具で巻き付けて配線するのですが，それに使われる線材がラッピングワイヤーです。ものがものだけにあまり少量では売ってはいませんが，1巻き買えば一生モノ(?)です。

の趣味で高密度実装に走っていますのであまり参考にはならないと思いますが。

抵抗などの数値は書き込んでいませんが回路図と見比べればすぐわかるでしょう。不親切なようですが，信号の流れを理解していただきたいのです。細かい配線は初心者の方には少々きついかもしれません。しかしいまこそ日本人の本領を発揮するときです。顕微鏡下で半田付けするプロの方に比べたら……。

図6の配線例ですが，電源ジャックの正負が，市販のACアダプタでは逆になっているものが多いようです（よく考えれば当たり前だね）。注意してください。

近頃の部品は自動工程に対応するため非常に丈夫です。よほどのことをしない限りはまず壊れませんので，落ち着いて余裕を持って作業してください。怪我をしたら元も子もありませんから。

完成の暁

できましたか？

まさかいきなりケースにまで組み込んじゃった人はいないでしょうね。まずは裸で動作を確認します。ICをソケットに挿す前にもう一度配線をチェックしてください。テスターをお持ちなら，まず抵抗レンジで電源の絶縁を確かめます（かといって絶縁計なんか持ち出しちゃあだめですよ）。一瞬，パスコンに充電電流が流れて，その後抵抗が∞になれば合格です。ではそのまま，電源を入れたX68000につないでみましょう。各ICの電源ピンに，正しく5Vがかかっているかどうかを確かめてください。大丈夫ならICを装着しましょう。くれぐれも方向を間違えないように。

動かしてみるのだ

一刻も早く動かしてみたいですね。でもデバッガでポートを眺めていても絵が見えてくるわけではありません。ソフトを用意する必要があります。リスト1にスキャナのドライバプログラムをX-BASICの外部関数のかたちで作ってみました。とりあえずこれだけで絵が取り込めます。scan( )でスキャナーからテキスト画面に画像を読み込み，ttog( )で画面モードにあわせて画像をグラフィック画面に転送します。詳しくはリスト中の説明をご覧ください。

プログラムの説明は割愛します。複雑なことはしていませんので……。時間と戦っているさまを笑ってやってください。なお，コンパイラまたはアセンブラからこれらの外部関数を使うことができます（ちょっと効率悪いですが）。その場合は，オブジェクトファイル(SCANNER.Oなど)をリンクしてください。外部関数のX-BASICへの組み込み方は皆さんご存じですよね？

ここで注意をひとつ。この読み取りプログラムは，スーパーバイザモードですべての割り込みを禁止して，最高速で突っ走るというマルチタスク破壊プログラムになっています。ですから暴走が怖い人でHuman68kのバージョン2をお使いの方は，CONFIG.SYSでPROCESSを指定しないでください。もしなにかあったときは，インタラプトスイッチを押して止めるしかないのです。

では再びアダプタをつなぎましょう。リスト2のテストプログラムを走らせてみます。スキャナの読み取りボタンを押しながら動かしてください。うまくいけば64mm×96mmの範囲が読み込まれるはずです。いかがですか？

ハングしてしまったら速やかにアダプタを抜き，もう一度配線なりプログラムなりをチェックしてください。

うまく動いたときの感動はハード工作ならではのものです。綺麗にケースに入れて本当の完成です。おめでとう。

10) BASICの外部関数を作るのにはアセンブラとリンカが必要です。福袋かCコンパイラを購入してください。

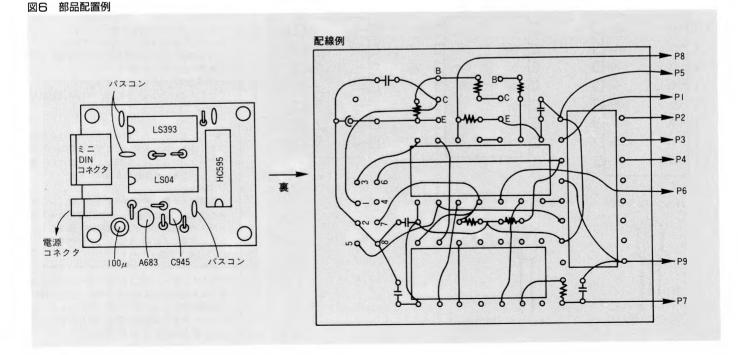
手順としては，ソースファイルをアセンブル・リンクして.Xファイルを作り，この拡張子を.FNCにリネームしてBASIC.Xのあるディレクトリにコピーします。そしてエディタでBASIC.CNFに，

FUNC＝ファイル名

の1行を追加して終わりです。SCANNER.Sの場合ですと，たとえば，

```
ED SCANNER.S
    (プログラムを打ち込む)
AS SCANNER
LK SCANNER
COPY SCANNER.X ¥BASIC2¥*.FNC
ED ¥BASIC2¥BASIC.CNF
  (FUNC = SCANNERを追加)
```

となります。

図6　部品配置例

ちょっと実用プログラム

さて，これだけでは物足りませんので，評価用の簡単なユーティリティを作ってみました（リスト3）。新たにリスト4の外部関数[10)]が必要です。できれば皆さんに使っていただきたいと思いましたので，これもBASICです。しかしX-BASICって強力ですね（トロいけど）。

それでは使い方を説明します。操作はマウスかキーボードで行います。マウスの上下，またはカーソルキーの↑↓で選択，左クリックまたはリターンかスペースキーで実行，右クリックまたはESCキーで取り消しです。

プログラムを走らせると，いきなり画面を真っ白に塗ります。実は，読み取り画像のバッファにテキスト画面の2，3プレーンを使っていて，このエリアは“終了”で“原画消去”を実行するまで消えません。よってなにか不都合が起きてプログラムが中断しても原画は生きていますので再実行すると表示されるようにしてあるのです。

次に，画面左上にメニューが現れます。それぞれの機能は見ればだいたい想像がつくと思いますが，一応順に説明します。

“画面モード”は，画面モードを変更します。512×512ドットモードを選ぶと16色か65536色かと聞かれますが，描画するモードを変更するだけで，表示されるイメージは同じです。

“取り込み”は，スキャナからの取り込みを実行します。ずらっと出てくる数字は，左から解像度，原稿の範囲，読み取る方向を表します（64㎜幅，200／400 dpiのスキャナを想定していますので，違う規格のスキャナをお使いの方は適当に変更してください）。読み取り速度には限界がありますので，あまり速くスキャンしないでください。

コラムにもあるとおり，原稿サイズと画面モードによってはオーバーサンプリングによる画像の多階調化を行っています。しかし，用途によっては完全な白黒の画像が必要なときもあります。そんな場合，画像の中間色に，あるしきい値を設けて白と黒の2値画像にするのが“2値化”です。画像を見ながら適当なしきい値を決めてください。

“保存”は画像をファイルに保存します。ファイル形式はおそらくX68000の標準的な形式であろうと思われる3種類をサポートしました。“.GS0”はimg_save( )の512×512，16色モードです。このときパレットを保存するパレットファイル（パレットコードが16個入っているだけですが）を一緒に作りますが，このファイルの拡張子はミュージカルプランのペイントソフトである「マジックパレット」で使われている形式（.PLS）としました。もちろん2値化してある場合にはパレットファイルは作りません。

“.PIC”はご存じPIC.Rによるファイル形式です。ここでは全画面モード対応のAPIC.Rをチャイルドプロセスで呼び出して使っていますので，APIC.Rがカレントパス上になければなりません（つまり，環境変数PATHのディレクトリ内のどこかにAPIC.Rを持ってくる必要がある）。APIC.Rは電波新聞社のゲームソフト「バブルボブル」にも入っていますから，お持ちの方も多いのではないかと思いますが，もちろんプログラムを少し変更することによって普通のPIC.Rを使うこともできます。思い切って本誌6月号のPIC.FNCを使ってもよいでしょう。

最後に“.CUT”はPDSのペイントソフトである「モノトーン」のファイルです。画面モード768×512で，2値化した画像のうち，680×480ドットが対象になります。このときの範囲指定はマウスでもカーソルキーでもできますが，標準のシステムではマウスの鈍さが目立ってしまい操作性がよくありません。勝手ながら電脳倶楽部に掲載されたMOUSE.X（マウス高速化ユーティリティ）との併用をおすすめします。

スキャナ用サンプル

以上3つの形式で保存することができますがこれらの形式以外でも，チャイルドプロセスを用いれば簡単にお手持ちのほかのグラフィックファイラーに対応させることができるでしょう。

“終了”は，プログラムを終了します。このとき“原画保存”を選択すると，テキスト画面のバッファをクリアしません。

残念ながらカラースキャナには対応していません。(5,000円で売ってたカラースキャナがあったのに……後悔先に立たず。）自走式ならただ3回読めばいいはずですから，簡単に対応できます。

また一歩野望に近づいた

そんなこんなで，永年の夢ハンディスキャナアダプタが完成しました。Z'sSTAFFでへらへら女の子の絵を描くもよし，文字認識に挑戦してダンプリストを読み込ませるもよし，自由に応用してやってください。

文中，一般受けしない専門用語がたくさん出てきたことをお詫びします。しかし，そんな言葉はわからなくて普通ですから気にしないでください。興味さえあればそのうち覚えます。おっと，もし貴方が工学系の学生ならこんな甘えは許されませんが。

最後になりましたが，ハードの動作確認に快く協力してくださった恒吉宏治氏，佐久間繁夫氏，並びに中村隆生氏に，この場をお借りしてお礼申しあげます。

リスト1

```
================  SCANNER.S  ==================
   1: **************************************************
   2: *
   3: *    X－BASIC拡張関数ライブラリ
   4: *        （XCライブラリ共通）
   5: *
   6: *    ハンディースキャナードライバ
   7: *
   8: *        SCANNER.FNC
   9: *
  10: *            Version 1.06
  11: * Presented by Hayashi Art Products  1990
  12: *
  13: **************************************************
  14:
  15: *==========---  R E F E R E N C E  ---==========*
  16: *
  17: *
  18: * S C A N
  19: *-------------------------------------------------*
  20: * (void) scan(port,plane,column,line ;int)
  21: *
  22: *        port  : ジョイスティックポート番号
  23: *                1 ～ 2
  24: *        plane : テキストプレーン番号
  25: *                0 ～ 3
  26: *        column: X方向バイト数 (1～128)
  27: *        line  : Y方向ライン数 (1～4096)
  28: *
  29: *   ハンディースキャナーからテキスト画面に
  30: * 画像データを取り込みます。
  31: *
  32: *
  33: *
```

```
34: * S P O W E R
35: *-----------------------------------------------*
36: * (void) spower(port,power ;int)
37: *
38: *        port  : ジョイスティックポート番号
39: *                1 ～ 2
40: *        power : 0 = 電源OFF
41: *                1 = 電源ON
42: *
43: *  ジョイスティックストローブ信号を用いて
44: * スキャナーの電源を制御します。
45: *
46: *
47: *
48: * T T O G
49: *-----------------------------------------------*
50: * (void) ttog(plane,mode,grayscale;int)
51: *
52: *        plane : テキストプレーン番号
53: *                0 ～ 2 ( 3 )
54: *        mode  : 転送モード(以下参照)
55: *        grayscale : 表示色を指定する
56: *                    int型配列名
57: *
58: *  モード:      転送サイズ          :中間調
59: *   0   : x768*y512  to  x512*y512 : 4 階調
60: *   1   : x1024*y512  to x1024*y512 : 2 階調
61: *   2   : x1024*y1536 to  y512*x512 : 7 階調
62: *   3   : x1024*y1536 to  y512*x768 : 5 階調
63: *
64: *   テキスト画面をグラフィック画面に圧縮
65: * 転送します。 各中間調はそれぞれgray-
66: * scale(0～6)で指定されるパレットコード
67: * になります。
68: *   テキストプレーン番号に3を指定できる
69: * のは転送モード0,1の時のみです。
70: *   エラーチェックは一切していません。画
71: * 面モード等をよく考えて、変な値を指定し
72: * ないようにして下さい。
73: *
74: *
75: *
76: *===============================================*
77:
78:  .include        FDEF.H
79:  .include        IOCSCALL.MAC
80:  .include        DOSCALL.MAC
81:
82:  .globl          _scan              *リンク用
83:  .globl          _spower
84:  .globl          _ttog
85:
86: *******  マクロ定義  *******
87:
88: PUSH      macro           REG_LIST
89:           movem.l         REG_LIST,-(sp)
90:           endm
91:
92: POP       macro           REG_LIST
93:           movem.l         (sp)+,REG_LIST
94:           endm
95:
96: *******  システムポート  *******
97:
98: GVRAM           equ             $C00000
99: GVLBH           equ             $CFF800
100: GVLBL          equ             $C7FC00
101: T0RAM          equ             $E00000
102:
103: PPIA           equ             $E9A001
104: PPIC           equ             $E9A005
105:
106: *                MSB                 LSB
107: *8255ポートAorB (IN)  = |1|SRB|SNX|1|D3|D2|D1|D0|
108:
109: * SRB      : データストローブ(両エッジ)
110: * SNX      : 副走査エンコーダー同期信号(負エッジ)
111: * D0～3    : 画像データ(0白/1黒, D3が先頭)
112:
113: *=================================================
114:
115: *******  X-BASIC用ヘッダ  *******
116:
117: init_adr        dc.l            return
118: run_adr         dc.l            return
119: end_adr         dc.l            return
120: system_adr      dc.l            return
121: break_adr       dc.l            return
122: input_adr       dc.l            return
123: reserve1        dc.l            return
124: reserve2        dc.l            return
125: token_adr       dc.l            token_table
126: param_adr       dc.l            param_table
127: exe_adr         dc.l            exe_table
128: reserve         ds.l            5
129:
130: token_table
131:                 dc.b            "scan",0
132:                 dc.b            "spower",0
133:                 dc.b            "ttog",0,0
134:
135:                 .even
136: param_table
137:                 dc.l            scan_param
138:                 dc.l            spower_param
139:                 dc.l            ttog_param
140:
141: scan_param
142:                 dc.w    int_val,int_val,int_val,int_val,void_ret
143: spower_param
144:                 dc.w    int_val,int_val,void_ret
145: ttog_param
146:                 dc.w    int_val,int_val,ary1_i,void_ret
147:
148: exe_table
149:                 dc.l            scan
150:                 dc.l            spower
151:                 dc.l            ttog
152:
153: *===============================================
154:                 .text
155:                 .even
156: return
157:                 rts
158:
159: *===============================================
160: *
161: * s c a n ( j , p , c , l )
162: *
163: *===============================================
164:                 .even
165: *
166: * X-BASICエントリ
167: *
168: scan
169:           move.l          12(sp),d0        *引数プッシュ
170:           move.l          22(sp),d1
171:           move.l          32(sp),d2
172:           move.l          42(sp),d3
173:           PUSH            d0-d3
174:
175:           bsr             _scan
176:
177:           lea.l           4*4(sp),sp
178:           moveq.l         #0,d0
179:           rts
180: *
181: * Cエントリ
182: *
183: _scan
184:           LINK            a6,#0
185:           PUSH            d1-d7/a0-a3
186:
187:           movem.l         8(a6),d0-d1      *引数取得
188:           move.l          d0,d2
189:           asl.l           #4,d2
190:           move.l          d2,a3            *a3=電源コントロールビット
191:           subq.l          #1,d0
192:           andi.l          #1,d0
193:           add.l           d0,d0
194:           lea.l           PPIA,a0
195:           adda.l          d0,a0            *a0=アクティブポートアドレス
196:           andi.l          #3,d1
197:           add.l           d1,d1
198:           swap.w          d1
199:           lea             T0RAM,a1
200:           adda.l          d1,a1            *a1=a2=テキストアドレス
201:           movea.l         a1,a2
202:           move.l          20(a6),d7        *d7=Yライン数
203:           subq.w          #1,d7
204:           move.b          #4,d4            *d4=シフトカウント(定数)
205:           move.b          #$20,d5          *d5=SNX信号チェック用(定数)
206:           move.b          #$C0,d6          *d6=SRB信号チェック用(定数)
207:
208:           clr.l           -(sp)            *スーパーヴァイザーモードに移行
209:           DOS             _SUPER
210:           move.l          d0,SSPBUF
211:           ori.w           #$0700,sr        *割り込み禁止
212:           move.w          a3,d0
213:           or.b            d0,PPIC          *電源ON
214:
215: s_newline
216:           move.l          20(a6),d3        *画面の一番下まで来たら
217:           subi.w          #513,d3          *上にスクロール
218:           sub.w           d7,d3
219:           blt             s_columnscan
220:           moveq.l         #8,d1
221:           moveq.l         #0,d2
222:           IOCS            _SCROLL
223: s_columnscan
224:           move.b          #$F,d2           *d2=データマスク用定数
225:           move.l          16(a6),d3        *d3=水平バイト数
226:           subq.w          #1,d3
227:
228: s_eloop1
229:           move.b          (a0),d0          *エンコーダー信号が
230:           and.b           d5,d0            *立ち上がるまで待つ
231:           bne             s_eloop1
232: s_eloop2
233:           move.b          (a0),d0
234:           and.b           d5,d0
235:           beq             s_eloop2
236:
237: s_skip11                                   *3ニブル読み飛ばす
238:           move.b          (a0),d0
239:           cmp.b           d6,d0
240:           bcs             s_skip11
241: s_skip12
242:           move.b          (a0),d0
243:           cmp.b           d6,d0
244:           bcc             s_skip12
245: s_skip13
```

▶引っ越しご苦労さまです。それと,うちの学校にはなんと動いているMZ-80K/2000/2200が合わせて40台ほどあって,いまでも授業で使ってますよ。　林崇洋(20)石川県

```
246:            move.b          (a0),d0
247:            cmp.b           d6,d0
248:            bcs             s_skip13
249:
250:                                            * - 実行時間(μs)
251: s_loop1                                       * ↓
252:            move.b          (a0),d0         *(>.8) 上位ニブル取り込み
253:            cmp.b           d6,d0           *( .4)
254:            bcc             s_loop1         *( .8)
255:            lsl.b           d4,d0           *(1.4) (Total>=3.4)
256: s_loop2
257:            move.b          (a0),d1         *(>.8) 下位ニブル取り込み
258:            cmp.b           d6,d1           *( .4)
259:            bcs             s_loop2         *( .8)
260:            and.b           d2,d1           *( .4)
261:            or.b            d0,d1           *( .4)
262:            move.b          d1,(a1)+        *( .8) 1バイト書き込む
263:            dbra            d3,s_loop1      *(1.0/1.4) (Total>=4.6)
264:
265:            lea             128(a2),a2      *次ラインアドレス
266:            movea.l         a2,a1
267:            dbra            d7,s_newline
268:
269:            move.w          a3,d0
270:            not.b           d0
271:            and.b           d0,PPIC         *電源OFF
272:            andi.w          #$f8ff,sr       *割り込み許可
273:            move.l          SSPBUF,(sp)     *ユーザーモード復帰
274:            DOS             _SUPER
275:            lea.l           4(sp),sp
276:            moveq.l         #8,d1           *HOME位置復帰
277:            moveq.l         #0,d2
278:            moveq.l         #0,d3
279:            IOCS            _SCROLL
280:
281:            POP             d1-d7/a0-a3
282:            UNLK            a6
283:            rts
284:
285: *=====================================================
286: *
287: * spower(port,ctrl)
288: *
289: *=====================================================
290:            .even
291: *
292: * X-BASICエントリ
293: *
294: spower
295:            move.l          12(sp),d0
296:            move.l          22(sp),d1
297:            PUSH            d0-d1
298:
299:            bsr             _spower
300:
301:            lea.l           4*2(sp),sp
302:            moveq.l         #0,d0
303:            rts
304: *
305: * Cエントリ
306: *
307: _spower
308:            LINK            a6,#0
309:            PUSH            d1-d2
310:            clr.l           -(sp)           *スーパーヴァイザーモード
311:            DOS             _SUPER
312:
313:            movem.l         8(a6),d1-d2
314:            asl.l           #4,d1
315:            andi.l          #1,d2
316:            beq             p_off
317:
318:            or.b            d1,PPIC         *電源ON
319:            bra             p_return
320: p_off
321:            not.b           d1              *電源OFF
322:            and.b           d1,PPIC
323: p_return
324:            move.l          d0,(sp)         *ユーザーモード復帰
325:            DOS             _SUPER
326:            lea.l           4(sp),sp
327:            POP             d1-d2
328:            UNLK            a6
329:            rts
330:
331: *==============================================
332: *
333: * ttog(plane,mode,gs)
334: *
335: *==============================================
336:            .even
337: *
338: * X-BASICエントリ
339: *
340: ttog
341:            move.l          12(sp),d0
342:            move.l          22(sp),d1
343:            move.l          32(sp),d2
344:            add.l           #10,d2          *配列要素先頭
345:            PUSH            d0-d2
346:
347:            bsr             _ttog
348:
349:            lea             4*3(sp),sp
350:            moveq.l         #0,d0
351:            rts
352: *
353: * Cエントリ
354: *
355: _ttog
356:            LINK            a6,#0
357:            PUSH            d1-d7/a0-a5
358:
359:            movem.l         8(a6),d0/d2/a5
360:            andi.w          #3,d0
361:            asl.w           #1,d0
362:            swap.w          d0
363:            lea             T0RAM,a0
364:            adda.l          d0,a0           *a0=テキストアドレス
365:            lea             GVRAM,a1        *a1=グラフィックアドレス
366:            andi.w          #3,d2           *a2=処理ルーチンアドレスを
   得る
367:            asl.w           #2,d2           *020ならこんなことしなくて
   もいい
368:            movea.l         TTOGTABLE(pc,d2.w),a2
369:
370:            clr.l           -(sp)
371:            DOS             _SUPER
372:            move.l          d0,SSPBUF
373:
374:            jsr             (a2)            *各モードごとの処理へ
375:
376:            move.l          SSPBUF,(sp)
377:            DOS             _SUPER
378:            lea.l           4(sp),sp
379:
380:            POP             d1-d7/a0-a3
381:            UNLK            a6
382:            rts
383:
384: TTOGTABLE                                   *逆アセ泣かせ
385: mode0add   dc.l            _mode0
386: mode1add   dc.l            _mode1
387: mode2add   dc.l            _mode2
388: mode3add   dc.l            _mode3
389:
390: *------------------------ MODE0,768*512 to 512*512,4colors
391: _mode0
392:            lea             VTABLE1,a2      *a2=明度テーブルアドレス
393:            move.w          #512-1,d7       *d7=ラインカウンター
394: t0_newline
395:            move.w          #32-1,d6        *d6=カラムカウンター
396: t0_columnloop
397:            moveq.l         #0,d0
398:            move.b          (a0)+,d0        *d0=テキストデータ(3バイ
   ト)
399:            lsl.l           #8,d0
400:            move.b          (a0)+,d0
401:            lsl.l           #8,d0
402:            move.b          (a0)+,d0
403:            lsl.l           #8,d0
404:
405:            move.w          #8-1,d5
406: t0_dotloop
407:            moveq.l         #0,d2
408:            rol.l           #3,d0           *d0から3ビット取り出しd1へ
409:            move.b          d0,d1
410:            andi.w          #7,d1
411:            add.w           d1,d1           *d1=d1*2
412:            move.b          0(a2,d1.w),d2   *d2=1ドット目輝度
413:            asl.b           #2,d2           *d2=d2*4
414:            move.w          2(a5,d2.w),(a1)+ *配列から色を得て画面へ
415:            move.b          1(a2,d1.w),d2   *以下2ドット目
416:            asl.b           #2,d2
417:            move.w          2(a5,d2.w),(a1)+
418:            dbra            d5,t0_dotloop
419:
420:            dbra            d6,t0_columnloop
421:
422:            lea.l           32(a0),a0       *次ラインのアドレス
423:            dbra            d7,t0_newline
424:
425:            rts
426:
427: *------------------------ MODE1,1024*512 to 1024*512,2colors
428: _mode1
429:            move.w          #32*512-1,d7    *d7=カウンター
430:            clr.w           d1
431:            move.w          #4,d2
432: t1_loop
433:            move.l          (a0)+,d0        *d0=テキストデータ
434:            rol.l           #2,d0
435:            move.w          #32-1,d6        *32ドット打つ
436: t1_dotloop
437:            rol.l           #1,d0
438:            move.b          d0,d1
439:            and.w           d2,d1           *d1=0 or 4
440:            move.w          2(a5,d1.w),(a1)+
441:            dbra            d6,t1_dotloop
442:
443:            dbra            d7,t1_loop
444:            rts
445:
446: *------------------------ MODE2,1024*1536 to 512*512,7colors
447: _mode2
448:            lea             VTABLE3,a2
449:            lea             GVLBL,a3        *a3=グラフィック左下のアド
   レス
450:            movea.l         a3,a1
451:            movea.w         #1024,a4        *a4=〃Y方向オフセット
452:            moveq.l         #0,d4
453:            move.w          #512-1,d7       *d7=ラインカウンター
454: t2_newline
455:            move.w          #64-1,d6        *d6=カラムカウンター
```

▶大野さんのあぶないシリーズ「ココムス」に対抗して，いろいろな表情をした（で）さんの顔が降ってくる「コムラス」というのはどうでしょう。絵は皆さんのご想像におまかせします。
清水健年(19)東京都

```
456: t2_columnloop
457:         move.w          256(a0),d1      *テキストのデータ、3ライン分
458:         move.w          128(a0),d2      *(d1-d3)
459:         move.w          (a0)+,d3
460:
461:         move.w          #8-1,d5         *8ドット打つ
462: t2_dotloop
463:         rol.w           #2,d1           *2*3=6ドットより1個点を打つ
464:         move.b          d1,d0
465:         andi.w          #3,d0
466:         move.b          (a2,d0.w),d4
467:
468:         rol.w           #2,d2
469:         move.b          d2,d0
470:         andi.w          #3,d0
471:         move.b          (a2,d0.w),d0
472:         add.b           d0,d4
473:
474:         rol.w           #2,d3
475:         move.b          d3,d0
476:         andi.w          #3,d0
477:         move.b          (a2,d0.w),d0
478:         add.b           d0,d4           *d4=輝度合計
479:
480:         asl.b           #2,d4
481:         move.w          2(a5,d4.w),(a1)
482:         suba.w          a4,a1           *一つ上の点のアドレス
483:         dbra            d5,t2_dotloop
484:
485:         dbra            d6,t2_columnloop
486:
487:         lea             256(a0),a0      *次ライン先頭のアドレス
488:         addq.l          #2,a3
489:         movea.l         a3,a1
490:
491:         dbra            d7,t2_newline
492:         rts
493:
494: *------------------------ MODE3,1024*1536 to 512*768,5colors
495: _mode3
496:         lea             VTABLE2,a2
497:         lea             GVLBH,a3        *a3=グラフィック左下アドレス
498:         movea.l         a3,a1
499:         movea.w         #2048,a4
500:
501:         move.w          #768-1,d7       *d7=ラインカウンター
502: t3_newline
503:         move.w          #64-1,d6        *d6=カラムカウンター
504: t3_columnloop
505:         move.w          128(a0),d1      *テキストのデータ、2ライン分
506:         move.w          (a0)+,d2
507:
508:         move.w          #8-1,d5         *8ドット打つ
509: t3_dotloop
510:         rol.w           #2,d1           *2*2=4ドットより1個点を打つ
511:         move.w          d1,d3
512:         andi.w          #3,d3
513:         asl.w           #2,d3
514:
515:         rol.w           #2,d2
516:         move.w          d2,d0
517:         andi.w          #3,d0
518:         or.w            d0,d3           *d3=4ドットのパターン
519:
520:         move.b          (a2,d3.w),d3
521:         asl.b           #2,d3
522:         move.w          2(a5,d3.w),(a1)
523:         suba.w          a4,a1
524:         dbra            d5,t3_dotloop
525:
526:         dbra            d6,t3_columnloop
527:
528:         lea             128(a0),a0      *次ラインのアドレス
529:         addq.l          #2,a3
530:         movea.l         a3,a1
531:
532:         dbra            d7,t3_newline
533:         rts
534:
535: *====================================================
536:         .data
537:         .even
538: ret_val_adr
539:         dc.w    0
540:         dc.l    0
541: return_val
542:         ds.l    1
543:
544: SSPBUF
545:         ds.l    1
546:
547: *******  ドットパターン→明度変換テーブル  *******
548:
549: VTABLE1                                 *3→2ドット(4階調)
550:         dc.b    0,0     * 000
551:         dc.b    0,2     * 001
552:         dc.b    1,1     * 010
553:         dc.b    1,3     * 011
554:         dc.b    2,0     * 100
555:         dc.b    2,2     * 101
556:         dc.b    3,1     * 110
557:         dc.b    3,3     * 111
558:
559: VTABLE2                                 *4→1ドット(5階調)
560:         dc.b    0       * 0000
561:         dc.b    1       * 0001
562:         dc.b    1       * 0010
563:         dc.b    2       * 0011
564:         dc.b    1       * 0100
565:         dc.b    2       * 0101
566:         dc.b    2       * 0110
567:         dc.b    3       * 0111
568:         dc.b    1       * 1000
569:         dc.b    2       * 1001
570:         dc.b    2       * 1010
571:         dc.b    3       * 1011
572:         dc.b    2       * 1100
573:         dc.b    3       * 1101
574:         dc.b    3       * 1110
575:         dc.b    4       * 1111
576:
577: VTABLE3                                 *2→1ドット(3階調)
578:         dc.b    0       * 00
579:         dc.b    1       * 01
580:         dc.b    1       * 10
581:         dc.b    2       * 11
582:
583: *====================================================
584:
585:         .end
```

リスト2

```
 1 int i,g(6)={31,28,24,19,14,8,0}
 2 screen 1,3,1,1
 3 console 0,32,0
 4 for i=0 to 5
 5   g(i)=hsv(0,0,g(i))
 6 next
 7 color [65534,0,0,0]
 8 print "Ready. Please Scan."
 9 scan(1,0,128,1536)
10 color [0,0,0,0]
11 ttog(0,2,g)
12 cls
13 color [0,65534,65534,65534]
14 end
```

リスト3

```
 1 /*--------------------------------------------------*/
 2 /*                 評価用プログラム                 */
 3 /*                                                  */
 4 /*           S C A N . B A S  Version 1.01          */
 5 /*                                                  */
 6 /*    ( 要・SCANNER.FNC,SCANSUPPORT.FNC,+α )       */
 7 /*                                                  */
 8 /*     presented by Hayashi Art Products 1990       */
 9 /*__________________________________________________*/
10 /*
11 int i,j,c1,c2,c3,curc,curs,white,tmp
12 int mpos=0,ch=0,act=0,ss=0
13 int JS=1  /* ジョイスティック端子番号 */
14 int srm=2 /* 画面モードデフォルト */
15           /*(0:512*512,16/ 1:512*512,65535/ 2:768*512) */
16 int scm=0 /* 原画サイズデフォルト */
17           /*(0:768*512/ 1:1024*1536/ 2:512*512/ 3:512*1536
) */
18 int XW(3)={96,128,64,64},YL(3)={512,1536,512,1536}
19 int GS(31),FULLCOL(255),cnvcol(6)
20 int PALSC(3,6)={ 0,31, 0, 0, 0, 0, 0
21                 ,0,14,24,31, 0, 0, 0
22                 ,0,11,19,26,31, 0, 0
23                 ,0, 8,14,19,24,28,31}  /*対数カーブになっ
ています
24 int CNVSC(5,6)={15, 0, 0, 0, 0, 0, 0
25                 ,3, 2, 1, 0, 0, 0, 0
26                 ,4, 3, 2, 1, 0, 0, 0
27                 ,6, 5, 4, 3, 2, 1, 0
28                ,31,24,14, 0, 0, 0, 0
29                ,31,28,24,19,14, 8, 0}
30 int imgbuf(5100)
31 char palbuf(32)
32 char MTHEAD(9)={2,0,0,2,168,1,224,0,0,0}
33 str message[100],mask[10]
34 /*
35 /*= MAIN MENU ===================================*/
36 /*
37 init()
38 flush()
39 cls
40 message="[画面モード][ 取り込み][ 二 値 化][  保  存][  終
了]["
41 mask="*****"
42 while 1
43   if curc=2 or ss>0 then mask[2]=32 else mask[2]=42
44   cls:tmp=menu(2,1,message,mask,ch)
45   if tmp=-1 then continue
46   ch=tmp
47   switch ch
48     case 0 :scrmod():break
49     case 1 :scanmod():break
```


▶編集部の中で「カノッサの屈辱」(フジテレビ)を見ている人いませんか？　私は初めて見たとき，オープニングと内容のギャップに大笑いしてしまいました。
岸村雅史(21)神奈川県

```
  50     case 2 :bicol():break
  51     case 3 :file():break
  52     case 4 :quit():break
  53   endswitch
  54 endwhile
  55 end
  56 /*
  57 /*= CHANGE SCREEN MODE ============================*/
  58 /*
  59 func scrmod()
  60  int tmp,sd=0
  61  str message[64]
  62  message="[ 5 1 2 × 5 1 2 ][ 7 6 8 × 5 1 2 ][ 取 り 消 し ][
"
  63  if srm=2 then sd=1
  64  tmp=menu(11,2,message,"***",sd)
  65  if tmp=-1 or tmp=2 then return()
  66  if tmp=0 then {
  67     message="[    1 6 色][ 6 5 5 3 5 色][  取り消し]["
  68     tmp=menu(23,3,message,"***",srm+2*(srm=2))
  69     if tmp=-1 or tmp=2 then return()
  70  } else {
  71    tmp=2
  72  }
  73  srm=tmp:ss=0
  74  waitmes():flush()
  75 endfunc
  76 /*
  77 /*- REDRAW SCREEN ------------------------------------*/
  78 /*
  79 func flush()
  80  wipe():home(0,0,0)
  81  if srm=0 then {          /* SWITCH文の中ではブロックIF文が
使えない */
  82    img_scrn(1,1,1):vpage(1)
  83    white=15
  84    if scm=2 then home(0,426,0)
  85    if scm=1 or scm=3 then {
  86      curc=7:curs=3:paldef():cnvdef()
  87      ttog(2,2,cnvcol)
  88    } else {
  89      curc=4:curs=1:paldef():cnvdef()
  90      ttog(2,0,cnvcol)
  91    }
  92    return()
  93  }
  94  if srm=1 then {
  95    img_scrn(1,2,1):curs=5:paldef():img_scrn(1,3,1)
  96    white=GS(31)
  97    if scm=2 then home(0,426,0)
  98    if scm=1 or scm=3 then {
  99      curc=7:cnvdef()
 100      ttog(2,2,cnvcol)
 101    } else {
 102      curc=4:curs=4:cnvdef()
 103      ttog(2,0,cnvcol)
 104    }
 105    return()
 106  }
 107  if srm=2 then {
 108    img_scrn(2,0,1)
 109    white=15
 110    if scm=2 then home(0,896,0)
 111    if scm=1 or scm=3 then {
 112      curc=5:curs=2:paldef():cnvdef()
 113      ttog(2,3,cnvcol)
 114    } else {
 115      curc=2:curs=0:paldef():cnvdef()
 116      ttog(2,1,cnvcol)
 117    }
 118  }
 119 endfunc
 120 /*
 121 /*- CONVERT_COLOR DEFINITION -------------------------*/
 122 /*
 123 func cnvdef()
 124  int i
 125  if ss=0 then {
 126    for i=0 to 6:cnvcol(i)=CNVSC(curs,i):next
 127  } else {
 128    for i=0 to curc-ss-1:cnvcol(i)=white:next
 129    for i=curc-ss to curc-1:cnvcol(i)=0:next
 130  }
 131 endfunc
 132 /*
 133 /*= EXECUTE SCANNING =================================*/
 134 /*
 135 func scanmod()
 136  int i,xw,yl
 137  str message[200],dns[16],dir[16]
 138  message="[400dpi:48×32mm ( ↓ )][400dpi:64×96mm ( → )][200
dpi:64×64mm ( ↓ )][200dpi:64×192mm( → )][     取 り 消 し]["
 139  tmp=menu(11,3,message,"*****",scm)
 140  if tmp=-1 or tmp=4 then return()
 141  scm=tmp
 142  cls:wipe():txtclr(3,0)
 143  color[GS(30),C1,C2,C3]
 144  if scm=2 or scm=3 then { txtclr(2,-1)
 145                           txtfill(2,0,0,512,512,&HAA55AA55
)
 146                           txtfill(3,512,0,512,512,-1)
 147                           dns="標準(200dpi)"
 148                    } else { txtclr(2,&HAA55AA55)
 149                           dns="高密度(400dpi)" }
 150  if scm=0 or scm=2 then { dir="上から下へ"
 151                    } else { dir="左から右へ" }
 152  mesprint(1,1," 読取密度を"+dns+"にセットし、"+dir+"読んで
下さい。")
 153  scan(JS,2,XW(scm),YL(scm))
 154  ss=0:color[0,C1,C2,C3]
 155  cls:waitmes():flush()
 156 endfunc
 157 /*
 158 /*= TO 2 COLORS ===================================*/
 159 /*
 160 func bicol()
 161  int i,tmp,srs=0
 162  str message[64]
 163  message=right$("[ 6 ][ 5 ][ 4 ][ 3 ][ 2 ][ 1 ][取消][",cu
rc*5+2)
 164  if srm=1 then srs=1:srm=0:waitmes():flush()
 165  act=1:cls
 166  tmp=menu(2,1,message,"*******",int(curc/2)-1)
 167  act=0
 168  if srs=1 then srm=1
 169  if (tmp=-1 or tmp=curc-1) then {
 170    if srs=0 then paldef():return()
 171  } else { ss=curc-tmp-1 }
 172  waitmes():flush()
 173 endfunc
 174 /*
 175 /*= SAVE FILE ======================================*/
 176 /*
 177 func file()
 178  int tmp
 179  str message[64],mask[10]=" * *"
 180  message="[.G S 3][.P I C][.C U T][ 取 消 ]["
 181  if srm=0 then mask[0]=42
 182  if srm=2 then mask[0]=42:if ss>0 or scm=0 or scm=2 then m
ask[2]=42
 183  switch menu(11,5,message,mask,0)
 184    case 0 :gs3save():break
 185    case 1 :picsave():break
 186    case 2 :cutsave()
 187  endswitch
 188 endfunc
 189 /*
 190 /*- .GS3 FILE SUPPORT -------------------------------*/
 191 /*
 192 func gs3save()
 193  int i,j,p,fp
 194  str fnm[32]
 195  if srm=2 and scm<>2 then txtfill(1,512,0,256,512,&HAA55AA
55)
 196  fnm=fnminput(14,12,".GS3")
 197  if fnm="" then return()
 198  if chkdskf(fnm,132100) then return()
 199  cls:mesprint(-1,15," "+fnm+".GS3 を保存しています。")
 200  img_save(fnm+".GS3")
 201  if ss>0 or curs=0 then return()
 202  fp=fopen(fnm+".PLS","c")
 203  if fp=-1 then fileerr("パレット"):return()
 204  cls:mesprint(-1,15," パレットファイル "+fnm+".PLS を保存
しています。")
 205  for i=0 to 31:palbuf(i)=0:next
 206  for i=0 to 6
 207    p=PALSC(curs,i)
 208    palbuf(i*2)=p ¥ 256
 209    palbuf(i*2+1)=p mod 256
 210  next
 211  fwrite(palbuf,32,fp)
 212  fclose(fp)
 213 endfunc
 214 /*
 215 /*- .PIC FILE SUPPORT -------------------------------*/
 216 /*                          A P I C・R をお持ちの方へ  */
 217 func picsave()
 218  int ec,fp
 219  str fnm[32]
 220  fnm=fnminput(14,12,".PIC")
 221  if fnm="" then return()
 222  cls:mesprint(-1,15," "+fnm+".PIC を保存しています。")
 223  ec=child("APIC /S "+fnm)
 224  if ec<0 then errmes(" A P I C・R の起動に失敗しました。")
:return()
 225  if ec>0 then fileerr("P I C"):return()
 226  console 0,31,0        /* APIC_G.Rは画面モードを復帰しない
 */
 227  fp=fopen("CON","w"):fwrites(chr$(27)+"[>5h",fp):fclose(fp
)
 228  mouse(2):mouse(4)
 229 endfunc
 230 /*
 231 /*- .CUT FILE SUPPORT -------------------------------*/
 232 /*                          モノトーンをお持ちの方へ   */
 233 func cutsave()
 234  int mx,my,bl,br,fp
 235  str k[1],fnm[32]
 236  cls:mouse(1):txthome(44,16)
 237  home(0,0,0)
 238  txtrev(1,86,30,684,2):txtrev(1,86,512,684,2)  /* 枠 */
 239  txtrev(1,86,32,2,480):txtrev(1,768,32,2,480)
 240  txtrev(0,86,30,683,1):txtrev(0,86,512,683,1)
 241  txtrev(0,86,31,1,481):txtrev(0,768,31,1,481)
 242  mesprint(13,3," 保存する範囲を指定して下さい ")
 243  msarea(0,0,88,32):setmspos(44,16)
 244  while 1
 245    msstat(mx,my,bl,br)
 246    if br or bl then kbfree()
 247    mspos(mx,my)
 248    k=inkey$(0):ak=asc(k)
 249    if ak then { /* 汚いけど高速化の為 */
 250      switch ak
 251        case 28 :mx=mx-(mx<88)-(mx<87):break
 252        case 29 :mx=mx+(mx>0)+(mx>1):break
 253        case 30 :my=my+(my>0)+(my>1):break
 254        case 31 :my=my-(my<32)-(my<31)
 255      endswitch
 256    }
 257    setmspos(mx,my)
```

▶今年の受験の鉄則は「ポピュラス」を買わないことです。　　野口友則(18)茨城県

```
 258    txthome(88-mx,32-my)
 259    if bl or ak=13 or ak=32 then fnm=fnminput(14,5,".CUT"):
break
 260    if br or ak=27 then break
 261   endwhile
 262   txtclr(0,0):txtclr(1,0):txthome(0,0)
 263   if fnm<>"" then if chkdskf(fnm,41000)=0 then {
 264     fp=fopen(fnm+".CUT","c")
 265     if fp=-1 then {
 266       fileerr("ＣＵＴ")
 267     } else {
 268       cls:mesprint(-1,15," "+fnm+".CUT を保存しています。")
 269       fwrite(MTHEAD,10,fp)
 270       hget(mx,my,85,240,imgbuf)
 271       fwrite(imgbuf,5100,fp)
 272       hget(mx,my+240,85,240,imgbuf)
 273       fwrite(imgbuf,5100,fp)
 274       fclose(fp)
 275     }
 276   }
 277   mouse(2)
 278   if scm=2 then home(0,896,0)
 279  endfunc
 280  /*
 281  /*- CHECK FREE_AREA SIZE ----------------------------*/
 282  /*
 283  func int chkdskf(fnm;str,size;int)
 284   int d,l,r
 285   if instr(1,fnm,":") then {
 286     d=fnm[0]-64
 287     if d>26 then d=d-32
 288   } else d=0
 289   if dskf(d)>=size then return(0)
 290   errmes(" ディスクの空き容量が足りません。")
 291   return(-1)
 292  endfunc
 293  /*
 294  /*- INPUT FILENAME ----------------------------------*/
 295  /*
 296  func str fnminput(xpos;int,ypos;int,ext;str)
 297   int xs,xx=0,ak,dmy,bl,br
 298   str fnm[20]="",k[1],mes[20]="ファイル名："
 299   mesprint(xpos,ypos,mes+space$(20)+ext)
 300   xs=xpos+12
 301   while 1
 302     locate xs+xx,ypos
 303     color 9:print" ";chr$(29);:color 3
 304     k=inkey$:ak=asc(k)
 305     switch ak
 306       case 8       /* 削除 */
 307       case 29
 308       case 127
 309         if xx=0 then break
 310         xx=xx-1
 311         color 11:print" ";:color 3
 312         fnm[xx]=0
 313         break
 314       case 13      /* 終了 */
 315       case 32
 316         color 11:print" ";:color 3
 317         if xx=0 then return("")
 318         mesprint(xpos+6,ypos+2," よろしいですか？［Ｙ／Ｎ］
")
 319         repeat
 320           k=inkey$(0)
 321           msstat(dmy,dmy,bl,br)
 322         until k<>"" or bl or br
 323         kbfree()
 324         if instr(1,"Yyﾝ",k) or bl then return(fnm)
 325       case 27
 326         return("")
 327       default
 328         if xx=19 or ak<32 then break
 329         color 5:print k;:color 3
 330         txtrev(1,(xs+xx)*8,ypos*16,8,16)
 331         txtrev(0,(xs+xx)*8,ypos*16,8,16)
 332         fnm[xx]=ak
 333         xx=xx+1
 334     endswitch
 335   endwhile
 336  endfunc
 337  /*
 338  /*= QUIT ==========================================*/
 339  /*
 340  func quit()
 341   switch menu(11,6,"[原画保存][原画消去][取り消し][","***",
0)
 342     case 0
 343       screen 2,0,1,0:palinit():spower(JS,0):end
 344     case 1
 345       txtclr(2,0):txtclr(3,0):width 96:spower(JS,0):end
 346   endswitch
 347  endfunc
 348  /*
 349  /*= CHOICE ITEM ON MENU ============================*/
 350  /*
 351  func int menu(x;int,y;int,message;str,mask;str,choice;int)
 352   int dx,dy,bl,br
 353   int i=0,yy=0,ps=2,dlm
 354   int l,ak,mmax,yl,mwidth=0,xp,xw,tmp
 355   str k[1],mes(10)
 356   xp=x*8
 357   while 1
 358     dlm=instr(ps,message,"][")
 359     if dlm=0 then break
 360     mes(i)=mid$(message,ps,dlm-ps)
 361     l=len(mes(i)):if l>mwidth then mwidth=l
 362     ps=dlm+2:i=i+1
 363   endwhile
 364   mmax=i-1
 365   yl=mmax*16+23
 366   xw=mwidth*8+18
 367   while 1
 368     if mask[choice]<>32 then break
 369     choice=choice+1
 370     if choice=mmax+1 then return(-1)
 371   endwhile
 372   txtfill(1,xp-3,y*16-3,xw+5,yl,0)
 373   txtfill(0,xp-3,y*16-3,xw+5,yl,0)
 374   for i=0 to mmax
 375     if mask[i]=32 then color 6 else color 5
 376     locate x+1,y+i:print mes(i)
 377   next
 378   color 3
 379   txtrev(1,xp-3,y*16-3,xw+5,yl)
 380   txtrev(0,xp-2,y*16-2,xw+2,yl-3)
 381   while 1
 382     txtrev(0,xp-1,(y+choice)*16-1,xw,18)
 383     action(choice)
 384     while 1
 385       k=inkey$(0):ak=asc(k)
 386       msstat(dx,dy,bl,br):yy=yy+dy
 387       if yy>5 then ak=31:yy=0
 388       if yy<-5 then ak=30:yy=0
 389       if ak+bl+br=0 then continue
 390       kbfree()
 391       if ak=13 or ak=32 or bl then return(choice)
 392       if ak=27 or br then return(-1)
 393       if ak=31 then { /* Down */
 394         txtrev(0,xp-1,(y+choice)*16-1,xw,18)
 395         tmp=choice /* Skip */
 396         while tmp<(mmax)
 397           tmp=tmp+1
 398           if mask[tmp]<>32 then choice=tmp:break
 399         endwhile
 400         break
 401       }
 402       if ak=30 then { /* Up */
 403         txtrev(0,xp-1,(y+choice)*16-1,xw,18)
 404         tmp=choice /* Skip */
 405         while tmp>0
 406           tmp=tmp-1
 407           if mask[tmp]<>32 then choice=tmp:break
 408         endwhile
 409         break
 410       }
 411     endwhile
 412   endwhile
 413  endfunc
 414  /*
 415  /*- ACTION AT CHANGING LINE ------------------------*/
 416  /*
 417  func action(ch;int)
 418   int i
 419   switch act
 420     case 1
 421       if ch=curc-1 then paldef():return()
 422       for i=0 to curc-ch-2:palet(i,0):next
 423       for i=curc-ch-1 to curc-1:palet(i,GS(31)):next
 424   endswitch
 425  endfunc
 426  /*
 427  /*= PRINT MESSAGE ==================================*/
 428  /*
 429  func mesprint(xpos;int,ypos;int,message;str)
 430   int x,y,l
 431   if xpos=-1 then xpos=32-(srm=2)*16-int(len(message)/2)
 432   x=xpos*8:y=ypos*16:l=len(message)*8
 433   locate xpos,ypos
 434   color 5:print message:color 3
 435   txtrev(1,x-2,y-2,l+5,21)
 436   txtrev(0,x-1,y-1,l+2,18)
 437  endfunc
 438  /*
 439  /*- WAIT A MOMENT ----------------------------------*/
 440  /*
 441  func waitmes()
 442   mesprint(-1,15," しばらくお待ち下さい。")
 443  endfunc
 444  /*
 445  /*- CANNOT CREATE FILE -----------------------------*/
 446  /*
 447  func fileerr(fnm;str)
 448   errmes(" "+fnm+" ファイルが作成できません。")
 449  endfunc
 450  /*
 451  /*- ERROR OCCURED! ---------------------------------*/
 452  /*
 453  func errmes(mes;str)
 454   int d,l,r
 455   cls:mesprint(-1,15,mes+"［確認］")
 456   beep
 457   repeat
 458     msstat(d,d,l,r)
 459   until inkey$(0)<>"" or d or l or r
 460   kbfree()
 461  endfunc
 462  /*
 463  /*= HOUSE KEEPING ==================================*/
 464  /*
 465  /*- INITIALIZE & DEFINE CONSTANTS ------------------*/
 466  /*
 467  func init()
 468   screen 2,0,1,1:console 0,32,0
 469   C1=hsv(0,0,26)  /* 非選択枝の色 */
 470   C2=hsv(0,18,12) /* 文字と枠の色 */
 471   C3=hsv(0,0,31)  /* 背景の色 */
 472   palinit()
```

▶最近，腹の調子が悪く，僕の腹の中でたくさんの人が沼にはまっていきます（ゴボシ，ゴボシ）。
福知健(19)京都府

```
473  mesprint(-1,15," Handy Scanner Driver & Filer v1.00  by
 .  .")
474  spower(JS,0)
475  mouse(0):mouse(2):mouse(4)
476  for i=0 to 31:GS(i)=hsv(0,0,i):next
477  for i=0 to 127
478    FULLCOL(i*2)=i*514+1
479    FULLCOL(i*2+1)=FULLCOL(i*2)
480  next
481  for i=0 to 3
482    for j=0 to 6
483      PALSC(i,j)=GS(PALSC(i,j))
484    next
485  next
486  for i=4 to 5
487    for j=0 to 6
488      CNVSC(i,j)=GS(CNVSC(i,j))
489    next
490  next
491 endfunc
492 /*
493 /*- TEXT PALET INITIALIZE ---------------------------*/
494 /*
495 func palinit()
496  int i
497  for i=0 to 3:txtpalet(i*4,0):next
498  for i=0 to 3:txtpalet(i*4+1,C1):next
499  for i=0 to 3:txtpalet(i*4+2,C2):next
500  for i=0 to 3:txtpalet(i*4+3,C3):next
501 endfunc
502 /*
503 /*- GRAPHIC PALET DEFINITION -----------------------*/
504 /*
505 func paldef()
506  int i
507  if curs>3 then {
508    for i=0 to 255:palet(i,FULLCOL(i)):next
509  } else {
510     for i=0 to 6:palet(i,PALSC(curs,i)):next
511     palet(15,GS(31))
512  }
513 endfunc
514 /*
515 /*- KEY & BUTTON FREE ? ----------------------------*/
516 /*
517 func kbfree()
518  int dd,dl,dr
519  repeat:until inkey$(0)=""
520  repeat:msstat(dd,dd,dl,dr):until dl+dr=0
521 endfunc
522 /*
523 /*= FIN ===========================================*/
524 /*
```

リスト4

```
================  SCANSUPPORT.S  ==================
  1: **************************************************
  2: *
  3: *    X－BASIC拡張関数ライブラリ
  4: *      （XCライブラリ共通）
  5: *
  6: *   スキャナードライバーサポート関数群
  7: *
  8: *        SCANSUPPORT.FNC
  9: *
 10: *          Version 1.03
 11: * Presented by Hayashi Art Products  1990
 12: *
 13: **************************************************
 14:
 15: *==========---  REFERENCE  ---========*
 16: *
 17: *
 18: * TXTCLR
 19: *----------------------------------------------*
 20: * (viod) txtclr(plane,style ;int)
 21: *
 22: *        plane : テキストプレーンナンバー
 23: *                0～3
 24: *        style : ペイントスタイル
 25: *                &hxx_xx_xx_xx
 26: *                  ↑       ↑
 27: *              1ライン目…4ライン目
 28: *
 29: *   テキスト画面をstyleで指定されるパターン
 30: * でクリアします。
 31: *
 32: *
 33: *
 34: * TXTPALET
 35: *----------------------------------------------*
 36: * (int) txtpalet(paletcode,colorcode ;int)
 37: *
 38: *        paletcode : テキストパレットコード
 39: *                    0～15
 40: *        colorcode : カラーコード
 41: *                    0～65535（－1）
 42: *
 43: *   テキストパレット（パレットブロック0）
 44: * を定義します。カラーコードが－1の時は
 45: * 現在のカラーコードを返します。
 46: *   _TPALET2(IOCS$14)を呼んでいるだけです
 47: * ので、XCでは必要ないでしょう。
 48: *
 49: *
 50: *
 51: * TXTHOME
 52: *----------------------------------------------*
 53: * (void) txthome(x,y ;int)
 54: *
 55: *          x,y : テキスト画面表示位置
 56: *
 57: *   テキスト画面をスクロールします。
 58: *
 59: *
 60: *
 61: * TXTFILL
 62: *----------------------------------------------*
 63: * (void) txtfill(plane,x1,y1,xw,yw,style ;int)
 64: *
 65: *        plane : テキストプレーンナンバー
 66: *                0～3
 67: *          x1  : 先頭X座標
 68: *          y1  : 先頭Y座標
 69: *          xw  : X方向幅
 70: *          yw  : Y方向幅
 71: *        style : ペイントスタイル
 72: *
 73: *   テキスト画面の任意の矩形領域を塗り潰し
 74: * ます。_TXFILLを呼ぶだけです。
 75: *
 76: *
 77: *
 78: * TXTREV
 79: *----------------------------------------------*
 80: * (void) txtrev(plane,x1,y1,xw,yw ;int)
 81: *
 82: *        plane : テキストプレーンナンバー
 83: *                0～3
 84: *          x1  : 先頭X座標
 85: *          y1  : 先頭Y座標
 86: *          xw  : X方向幅
 87: *          yw  : Y方向幅
 88: *
 89: *   テキスト画面の任意の矩形領域を反転
 90: * します。これも_TXREVを呼ぶだけです。
 91: *
 92: *
 93: *
 94: *   ～～ .CUTファイルサポート用 ～～
 95: *
 96: * HGET
 97: *----------------------------------------------*
 98: * (void) hget(x1,y1,xw,yw ;int, ary ;ary1_fic)
 99: *
100: *          x1 : 始点X座標
101: *          y1 : 始点Y座標
102: *          xw : X方向カラム数
103: *          yw : Y方向ライン数
104: *         ary : 数値型一次元配列名
105: *
106: *   実画面サイズ1024*1024のグラフィック
107: * 画面を、1ドット1ビットで配列に読み
108: * 込みます。（データは4プレーンのOR）
109: *   X方向の幅はキャラクタ単位の指定で
110: * すので注意して下さい。
111: *
112: *
113: *
114: *   ～～ 外部ファイラー対応用 ～～
115: *
116: * CHILD                                        *
117: *----------------------------------------------
118: * (int) child(cli ;str)
119: *
120: *          cli : コマンドライン文字列
121: *
122: *   チャイルドプロセスを起動します。終了
123: * コードを返します。
124: *   Oh！X90年1月号，X68kマシン
125: * 語プログラミング《入門編》より拝借いた
126: * しました。
127: * （なんで！でできないんだー）
128: *
129: *
130: *==============================================*
131:
132:    .include        FDEF.H
133:    .include        IOCSCALL.MAC
134:    .include        DOSCALL.MAC
135:
136:    .globl          _txtclr          *リンク用
137:    .globl          _txtpalet
138:    .globl          _txthome
139:    .globl          _txtfill
140:    .globl          _txtrev
141:    .globl          _hget
142:    .globl          _child
143:
144: *******  マクロ定義  *******
145:
146: PUSH       macro          REG_LIST
147:            movem.l        REG_LIST,-(sp)
148:            endm
149:
```

▶ポピュラス本体では空飛ぶじゅうたん，プロミストランドではカニの化け物。マップを横断しながら山脈を作っていくこいつらは何者なんでしょう？　中花也須史(22)神奈川県

```
150: POP        macro           REG_LIST
151:            movem.l         (sp)+,REG_LIST
152:            endm
153:
154: *******  ＶＲＡＭアドレス *******
155:
156: T0RAM              equ             $E00000
157: GVRAM              equ             $C00000
158:
159: *******  Ｘ－ＢＡＳＩＣ用ヘッダ  *******
160:
161: init_adr           dc.l            return
162: run_adr            dc.l            return
163: end_adr            dc.l            return
164: system_adr         dc.l            return
165: break_adr          dc.l            return
166: input_adr          dc.l            return
167: reserve1           dc.l            return
168: reserve2           dc.l            return
169: token_adr          dc.l            token_table
170: parameter_adr      dc.l            param_adr_table
171: exe_adr            dc.l            exe_table
172: reserve            ds.l            5
173:
174: token_table
175:            dc.b            "txtclr",0
176:            dc.b            "txtpalet",0
177:            dc.b            "txthome",0
178:            dc.b            "txtfill",0
179:            dc.b            "txtrev",0
180:            dc.b            "hget",0
181:            dc.b            "child",0,0
182:
183:            .even
184: param_adr_table
185:            dc.l            txtclr_param_table
186:            dc.l            txtpalet_param_table
187:            dc.l            txthome_param_table
188:            dc.l            txtfill_param_table
189:            dc.l            txtrev_param_table
190:            dc.l            hget_param_table
191:            dc.l            child_param_table
192:
193: txtclr_param_table
194:            dc.w    int_val,int_val,void_ret
195: txtpalet_param_table
196:            dc.w    int_val,int_val,int_ret
197: txthome_param_table
198:            dc.w    int_val,int_val,void_ret
199: txtfill_param_table
200:            dc.w    int_val,int_val,int_val,int_val,int_val,int_val,void_ret
201: txtrev_param_table
202:            dc.w    int_val,int_val,int_val,int_val,int_val,void_ret
203: hget_param_table
204:            dc.w    int_val,int_val,int_val,int_val,ary1_fic,void_ret
205: child_param_table
206:            dc.w    str_val,int_ret
207:
208: exe_table
209:            dc.l            txtclr
210:            dc.l            txtpalet
211:            dc.l            txthome
212:            dc.l            txtfill
213:            dc.l            txtrev
214:            dc.l            hget
215:            dc.l            child
216:
217: *==================================================
218:            .text
219:            .even
220: return
221:            rts
222:
223: *==================================================
224: *
225: * ｔｘｔｃｌｒ（ｐｌａｎｅ，ｓｔｙｌｅ）
226: *
227: *==================================================
228:            .even
229: *
230: * X-BASICエントリ
231: *
232: txtclr
233:            move.l          12(sp),d0       *引数プッシュ
234:            move.l          22(sp),d1
235:            PUSH            d0-d1
236:
237:            bsr             _txtclr
238:
239:            lea             4*2(sp),sp
240:            moveq.l         #0,d0
241:            rts
242: *
243: * Cエントリ
244: *
245: _txtclr
246:            LINK            a6,#0
247:            PUSH            d1-d3/a0
248:
249:            movem.l         8(a6),d0/d1
250:            andi.l          #3,d0
251:            moveq.l         #1,d3
252:            lsl.w           d0,d3           *d3=コピープレーン指定
253:            asl.w           #1,d0
254:            swap.w          d0
255:            lea             T0RAM,a0
```

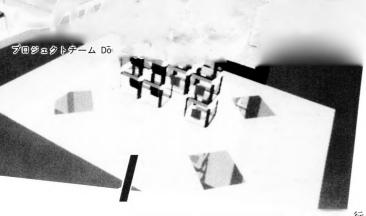
```
256:            adda.l          d0,a0           *a0=テキストアドレス
257:
258:            clr.l           -(sp)
259:            DOS             _SUPER
260:
261:            move.w          #128-1,d2       *とりあえず１ラスタをクリア
262:            rol.l           #8,d1
263: tcloop1:   move.b          d1,(a0)+
264:            dbra            d2,tcloop1
265:
266:            moveq.l         #128-1,d2
267:            rol.l           #8,d1
268: tcloop2:   move.b          d1,(a0)+
269:            dbra            d2,tcloop2
270:
271:            moveq.l         #128-1,d2
272:            rol.l           #8,d1
273: tcloop3:   move.b          d1,(a0)+
274:            dbra            d2,tcloop3
275:
276:            moveq.l         #128-1,d2
277:            rol.l           #8,d1
278: tcloop4:   move.b          d1,(a0)+
279:            dbra            d2,tcloop4
280:
281:            move.l          d0,(sp)
282:            DOS             _SUPER
283:            lea.l           4(sp),sp
284:
285:            move.w          #$0001,d1       *第０ラスタ→第１ラスタ
286:            move.w          #256,d2         *２５６ラスタを
287:            IOCS            _TXRASCPY       *下方向にコピー
288:
289:            POP             d1-d3/a0
290:            UNLK            a6
291:            rts
292:
293: *==================================================
294: *
295: * ｔｘｔｐａｌｅｔ（ｐａｌｅｔ，ｃｏｌｏｒ）
296: *
297: *==================================================
298:            .even
299: *
300: * X-BASICエントリ
301: *
302: txtpalet
303:            move.l          12(sp),d0
304:            move.l          22(sp),d1
305:            PUSH            d0-d1
306:
307:            bsr             _txtpalet
308:
309:            lea.l           4*2(sp),sp
310:            move.l          d0,return_val   *返り値
311:            moveq.l         #0,d0
312:            lea.l           ret_val_adr(pc),a0
313:            rts
314: *
315: * Cエントリ
316: *
317: _txtpalet
318:            LINK            a6,#0
319:            PUSH            d1-d2
320:
321:            movem.l         8(a6),d1-d2
322:            andi.l          #$F,d1
323:            IOCS            _TPALET2
324:
325:            POP             d1-d2
326:            UNLK            a6
327:            rts
328:
329: *==================================================
330: *
331: * ｔｘｔｈｏｍｅ（ｘ，ｙ）
332: *
333: *==================================================
334:            .even
335: *
336: * X-BASICエントリ
337: *
338: txthome
339:            move.l          12(sp),d0
340:            move.l          22(sp),d1
341:            PUSH            d0-d1
342:
343:            bsr             _txthome
344:
345:            lea.l           4*2(sp),sp
346:            moveq.l         #0,d0
347:            rts
348: *
349: * Cエントリ
350: *
351: _txthome
352:            LINK            a6,#0
353:            PUSH            d1-d3
354:
355:            moveq.l         #8,d1
356:            movem.l         8(a6),d2-d3
357:            IOCS            _SCROLL
358:
359:            POP             d1-d3
360:            UNLK            a6
361:            rts
```

```
362:
363: *==================================================
364: *
365: * t x t f i l l ( p , x 1 , x 2 , x w , y w , s )
366: *
367: *==================================================
368:         .even
369: *
370: * X-BASICエントリ
371: *
372: txtfill
373:         move.l          12(sp),d0
374:         move.l          22(sp),d1
375:         move.l          32(sp),d2
376:         move.l          42(sp),d3
377:         move.l          52(sp),d4
378:         move.l          62(sp),d5
379:         PUSH            d0-d5
380:
381:         bsr             _txtfill
382:
383:         lea.l           4*6(sp),sp
384:         moveq.l         #0,d0
385:         rts
386: *
387: * Cエントリ
388: *
389: _txtfill
390:         LINK            a6,#0
391:         PUSH            d1-d5/a1
392:
393:         movem.l         8(a6),d0-d5     *ロングワード列を
394:         movem.w         d0-d5,-(sp)     *ワード列にする
395:         movea.l         sp,a1
396:         IOCS            _TXFILL
397:         lea.l           2*6(sp),sp
398:
399:         POP             d1-d5/a1
400:         UNLK            a6
401:         rts
402:
403: *==================================================
404: *
405: * t x t r e v ( p , x 1 , x 2 , x w , y w )
406: *
407: *==================================================
408:         .even
409: *
410: * X-BASICエントリ
411: *
412: txtrev
413:         move.l          12(sp),d0
414:         move.l          22(sp),d1
415:         move.l          32(sp),d2
416:         move.l          42(sp),d3
417:         move.l          52(sp),d4
418:         PUSH            d0-d4
419:
420:         bsr             _txtrev
421:
422:         lea.l           4*5(sp),sp
423:         moveq.l         #0,d0
424:         rts
425: *
426: * Cエントリ
427: *
428: _txtrev
429:         LINK            a6,#0
430:         PUSH            d1-d4/a1
431:
432:         movem.l         8(a6),d0-d4
433:         movem.w         d0-d4,-(sp)
434:         movea.l         sp,a1
435:         IOCS            _TXREV
436:         lea.l           2*5(sp),sp
437:
438:         POP             d1-d4/a1
439:         UNLK            a6
440:         rts
441:
442: *==================================================
443: *
444: * h g e t ( x 1 , x 2 , x w , y w , a r y )
445: *
446: *==================================================
447:         .even
448: *
449: * X-BASICエントリ
450: *
451: hget
452:         move.l          12(sp),d0
453:         move.l          22(sp),d1
454:         move.l          32(sp),d2
455:         move.l          42(sp),d3
456:         move.l          52(sp),d4
457:         add.l           #10,d4          *配列先頭
458:         PUSH            d0-d4
459:
460:         bsr             _hget
461:
462:         lea.l           4*5(sp),sp
463:         moveq.l         #0,d0
464:         rts
465: *
466: * Cエントリ
467: *
468: _hget
469:         LINK            a6,#0
470:         PUSH            d1-d6/a0-a2
471:
472:         movem.l         8(a6),d1-d4/a2
473:         lea.l           GVRAM,a0
474:         add.l           d1,d1
475:         adda.l          d1,a0
476:         moveq.l         #11,d0
477:         asl.l           d0,d2
478:         adda.l          d2,a0           *a0=グラフィックアドレス
479:         movea.l         a0,a1
480:         subq.w          #1,d4           *d4=ラインカウンタ
481:         subq.w          #1,d3           *d3=カラム数
482:
483:         clr.l           -(sp)
484:         DOS             _SUPER
485: h_newline
486:         move.w          d3,d5           *d5=カラムカウンタ
487: h_columloop
488:         move.w          #8-1,d6         *d6=ドットカウンタ
489:         move.w          #0,d1
490: h_dotloop
491:         lsl.b           #1,d1
492:         tst.w           (a0)+
493:         bne             h_itsw          *白かな?
494:         addq.b          #1,d1
495: h_itsw
496:         dbra            d6,h_dotloop
497:
498:         move.b          d1,(a2)+
499:         dbra            d5,h_columloop
500:
501:         lea.l           2048(a1),a1     *1ライン下
502:         movea.l         a1,a0
503:         dbra            d4,h_newline
504:
505:         move.l          d0,(sp)
506:         DOS             _SUPER
507:         lea.l           4(sp),sp
508:
509:         POP             d1-d6/a0-a2
510:         UNLK            a6
511:         rts
512:
513: *==================================================
514: *
515: * c h i l d ( c l i )
516: *
517: *==================================================
518:         .even
519: *
520: * X-BASICエントリ
521: *
522: child
523:         move.l          12(sp),-(sp)
524:
525:         bsr             _child
526:
527:         lea.l           4(sp),sp
528:         move.l          d0,return_val   *返り値
529:         moveq.l         #0,d0
530:         lea.l           ret_val_adr(pc),a0
531:         rts
532: *
533: * Cエントリ
534: *
535: _child
536:         LINK            a6,#-512
537:         PUSH            d1-d7/a0-a6
538:
539:         movea.l         8(a6),a1
540:         lea.l           -512(a6),a0     *ファイル名を入れるところ
541:         move.w          #255-1,d0
542: _c_strcpy
543:         move.b          (a1)+,(a0)+
544:         dbeq            d0,_c_strcpy
545:         clr.b           (a0)
546:
547:         clr.l           -(sp)
548:         pea.l           -256(a6)        *その他を入れるところ
549:         pea.l           -512(a6)
550:         move.w          #2,-(sp)        *どこにいるかな?
551:         DOS             _EXEC
552:         tst.l           d0
553:         bmi             _c_exit
554:
555:         clr.w           (sp)            *実行
556:         DOS             _EXEC
557: _c_exit
558:         lea.l           14(sp),sp
559:
560:         POP             d1-d7/a0-a6
561:         UNLK            a6
562:         rts
563:
564: *==================================================
565:         .data
566:         .even
567: ret_val_adr
568:         dc.w    0
569:         dc.l    0
570: return_val
571:         ds.l    1
572:
573: *==================================================
574:
575:         .end
```

▶マジックバス! オークスター! なつかし〜。1982年12月号の表紙には見覚えがあります。あの頃はハドソンのゲームのリストが掲載してありましたね。創刊100号, おめでとうございます。
船山竜士(21)埼玉県

現

プロジェクトチーム Dō

今回は，マニュアルにも載っていない高度なテクニックの第１弾ということで，表現法をいくつか取り上げました。本誌のGraphic Galleryのサンプル画像と並行してご覧ください。

行った本連載へのアンケートを集計してみると，意外にも「マニュアルにも載っていない高度なテクニック」を希望する声が最も多くありました。そこまでCGAシステムを使いこなしているとは到底思いません（失礼！）が，知っていて損はないし，皆さんの声を反映させるためにも，さっそくさまざまなテクニックをまとめてみました。

映り込み

まずは，非常に古典的な手法で，床の映り込み（21ページ・画像１）です。鏡面などのツルツルな表面への映り込みは，透明体の屈折と並んでレイトレースの十八番であり，映り込みさえしていれば，パソコンCGとして一人前といった時代もありました。CGAシステムのレンダリングツールであるRENDでは，スキャンラインというレイトレースより高速性を優先したアルゴリズムを用いているため，基本的に映り込みを表現することはできません。ですから，この手法は映り込んでいるかのようにごまかすテクニックといえます。

図１をご覧ください。まず，a)は物体が床に映り込んでいる様子です。b)はa)から，映り込みはそのままにして物体だけを除いた図です。そしてc)は，半透明の床の下に上下逆さまにした物体を置いた図です。よく見ると，c)の床から透けて見える部分と，b)の床に映っている部分は同じであることがわかります。CGAシステムでは半透明体をサポートしていますので，このような画像は簡単にできます。つまり，映り込みは，半透明の床の下に，上下逆さまにした物体を置くことでそれらしく見

リスト１　映り込みのフレームファイル

```
env { back { rgb ( 0.00 0.20 0.50 ) } }
/************************ ﾏｽﾀｰ *************************************/
fram {
        { mov (     1100    -1400      800  ) eye deg( 60 )        }
        { mov (      600     -400      250  )  target              }
        light pal(  1  -3 -2 -4 )
        { mov ( 600 -400 0 )                obj     tet
                                            obj     kabe1
                                            obj     kabe2
                                            obj     yuka

        }
}
/************************ ﾋﾀﾞﾘ ｼﾀ ｵｸ *********************************/
fram  {
        { mov (     1100    -1400      800  ) eye deg( 60 )        }
        { mov (      600     -400      250  )  target              }
      * scal ( -1 -1 -1 )
        light pal(  1  -3 -2 -4 )
        { mov ( 600 -400 0 )                obj     tet
            }
}
/************************ ﾋﾀﾞﾘ ｼﾀ    *********************************/
fram  {
        { mov (     1100    -1400      800  ) eye deg( 60 )        }
        { mov (      600     -400      250  )  target              }
        scal ( -1  1 -1 )
        light pal(  1  -3 -2 -4 )
        { mov ( 600 -400 0 )                obj       tet
                                            obj     kabe1
        }
}
/************************ ﾋﾀﾞﾘ     ｵｸ *********************************/
fram  {
        { mov (     1100    -1400      800  ) eye deg( 60 )        }
        { mov (      600     -400      250  )  target              }
        scal ( -1 -1  1 )
        light pal(  1  -3 -2 -4 )
        { mov ( 600 -400 0 )                obj     tet
                                            obj     yuka
        }
}
/************************        ｼﾀ ｵｸ *********************************/
fram  {
        { mov (     1100    -1400      800  ) eye deg( 60 )        }
        { mov (      600     -400      250  )  target      }
        scal (  1 -1 -1 )
        light pal(  1  -3 -2 -4 )
        { mov ( 600 -400 0 )                obj     tet
                                            obj     kabe2
        }
}
/************************           ｵｸ *********************************/
fram  {
        { mov (     1100    -1400      800  ) eye deg( 60 )        }
        { mov (      600     -400      250  )  target              }
        scal (  1 -1  1 )
        light pal(  1  -3 -2 -4 )
        { mov ( 600 -400 0 )                obj     tet
                                            obj     kabe2
                                            obj     yuka
        }
}
/************************        ｼﾀ     ********************************/
fram  {
        { mov (     1100    -1400      800  ) eye deg( 60 )        }
        { mov (      600     -400      250  )  target              }
        scal (  1  1 -1 )
        light pal(  1  -3 -2 -4 )
        { mov ( 600 -400 0 )                obj     tet
                                            obj     kabe1
                                            obj     kabe2
        }
}
/************************ ﾋﾀﾞﾘ          ********************************/
fram  {
        { mov (     1100    -1400      800  ) eye deg( 60 )        }
        { mov (      600     -400      250  )  target              }
        scal ( -1  1  1 )
        light pal(  1  -3 -2 -4 )
        { mov ( 600 -400 0 )                obj     tet
                                            obj     kabe1
                                            obj     yuka

        }
}

* 注意
        yuka  : -600 ≦ x ≦ 600    -400 ≦ y ≦ 400   z = 0
        tet   : 0 ≦ z ≦ 700
        kabe1 : y = 400
        kabe2 : x = -600



= = = =   映り込みのバッチファイル
rend mirr.frm *.suf mirr.atr /a2 /c512 /tmirr /s2:2
rend mirr.frm *.suf mirr.atr /a2 /c512 /tmirr /s3:3 /hmirr002
rend mirr.frm *.suf mirr.atr /a2 /c512 /tmirr /s1:4 /hmirr003
        . . .
rend mirr.frm *.suf mirr.atr /a2 /c512 /tmirr /s8:8 /hmirr007
rend mirr.frm *.suf mirr.atr /a2 /c512 /tmirr /s1:1 /hmirr008
```

せているだけなのです。d)のように，不要な部分を不透明な面で隠せば，a)とまったく同じ絵になります。同様に，e)のように半透明の壁を作り，その向こうに左右逆さまの物体を置くことで，鏡のような効果を出すことも簡単です。

さらにそれらしく見せるために，もう少し小細工を施しましょう。まず，床の場合，ただ1面の板では気分が出ませんので，チェッカーボードのように模様を入れてやります。このときのアトリビュートですが，透明度は0.3ぐらいが適当でしょう。そして，模様の中に1種類だけ0.5ぐらいの面を混ぜておくと，そこだけ周りよりはっきりと映り込みをし，床の材質感がリアルになります。

鏡の場合，床よりも透明度を高くして（0.7ぐらい），やや青みがかった色にするのがよいでしょう。さらに芸の細かい話ですが，半透明の壁にf)のような模様を入れておくと，もっと鏡っぽくなります。この模様はなんなんだと聞かれると困るのですが，日本の漫画やアニメでは，なぜか鏡にはこんなスジを入れることになっているのです。

これらのテクニックだけでも十分に人をだますことができるのですが，よく考えるとウソがあります。g)のように，本来映り込みの物体は，光の当たり方，陰影のつき方も上下逆になるはずです。しかし，床の下に物体を置いただけでは，h)のようになってしまいます。これを解決するのも簡単で，まず，光源の位置や方向も上下逆にして，b)のような画像を作画し，あとで物体だけの画像と合成するのです（最新版のRENDでは，合成しながら作画することができます）。さらに，位置関係をよく考えれば，i)や画像1のように映り込みの映り込みなんて器用なこともできます。画像1のフレームファイルと，RENDの実行の仕方などをリスト1にまとめますので，じっくり考えてみてください。

しかしこれらのテクニックは，しょせんはごまかしですので，曲面への映り込みなんてできるわけがない……ことはありません。その曲面から反射方向に見た画像をあらかじめ計算しておいて，その画像をその曲面にマッピングすればよいのです。もちろん，やったことはありませんが……。

空気遠近法

これは，ただいまバージョンアップサービスを行っている最新版のRENDに新しく加わった機能で，手軽なわりに効果はそこそこあります。

画像2と画像3（21ページ）をご覧ください。空気遠近法とは，遠くの建物や山々の大気によるかすみを表現して，奥行きを強調する手法です。画像2が従来のもので，遠くの物もはっきり鮮やかなのに対して，空気遠近法を用いている画像3では，遠くの物ほど青白くかすんでいるのがよくわかるでしょう。CGらしい鮮やかさは失うものの，リアリティ，存在感というものが出ていると思います。

アルゴリズムは非常に簡単で，プログラムの改造も，あっという間でした。CGで隠面消去の計算をするときには，必ず視点からその物体（面）までの距離を求めます。そこで，色を計算するときに，その距離に応じた大気の色を加えてやるだけでいいのです。作画速度もほとんど変わりません。しかし，決して霧による散乱などを計算しているわけではないので，夜霧の中に浮かぶぼやけた街頭の灯などを表現することはできません。

さて，具体的な使用法ですが，今回配布するFFEはちゃんと空気遠近法に対応しており，背景設定の中で指示どおりデータを入れていただくだけで設定できます。与えるデータは，〈距離〉と〈色〉2種類です。〈色〉は上

図1　映り込み

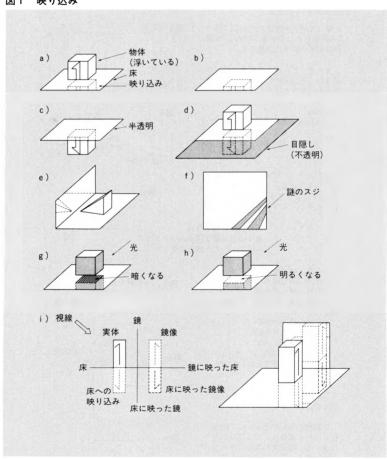



図2　空気遠近法の距離データ

記にもあった大気の色であり，遠くの物体がどんな色に染まっていくのかを，RGBで与えます。空の色というのは，ある意味で大気の色といえますので，通常この値は背景の色と同じになります。画像３ではRGB＝0.2, 0.7, 1.0としています。ただし，夕焼けとか夜とかの場合はまた異なる色を設定するべきでしょう。

そしてちょっと難しいのが〈距離〉のデータです。この〈距離〉がたとえば1000とすると，視点から1000離れた物体は，物体独自の色に大気の色が50％の割合で合成されます。では，2000離れた物体は100％大気の色になってしまうかというとそうではなく，1000からさらに1000離れているので，50％のさらに50％（つまり75％）が大気の色となり，物体独自の色は25％になってしまうのです。ですから，この値よりも視点に近い物体は，あまり空気遠近法の影響がなく，この値の２倍よりも遠い物体は，かすんではっきり見えないといえます。つまり，この値が小さいとスモッグの濃度が濃くなるわけです。図２を参考に値を決めてください。

FFEなど使わずに，フレームソースをエディタで制作するパワーユーザーのために，フレームソースの書式を記します。空気遠近法のデータは，環境文（env {…}）で指定します。

```
    depth    (<距離><色>)
```

画像３の例では，

```
    env {  back (          rgb ( 0.2  0.7  1.0 ) )
           depth ( 2500    rgb ( 0.2  0.7  1.0 ) )
        }
```

となっています。

さて，この手法には意外なメリットがあります。遠くのほうはかすんで見えなくなるので，背景まで細かくつくる必要がないことです。画像２では，いちばん奥の建物が，ただの直方体（あるいはただの板）であることがすぐばれてしまいますが，画像３では，なんら違和感がありません。大きな都市をデザインする場合は，遠景など，ビル街の影の形をした板を１枚用意するだけでよくなります。

しかし，手軽に遠近感が強調できるからといって，かたっぱしから利用するのはいけません。たとえば，自分の部屋の中をシミュレートするのに使っても不自然になるばかりです。もっとも，自分の部屋がドーム球場より広いとか，火事で煙が充満しているところを描いてみたいという趣味でしたらご自由に。

マッピング

大部分の方がご存じのことと思いますが，マッピングとは，木目や大理石の模様の画像データを３次元物体の表面に貼り付け，リアルな材質感を得る手法です。アンケートでもマッピングの使用法を公開してほしいという要望をたくさんいただきました。現在のCGAシステムでは，マッピングに関するツールがまったくなく，操作が非常に難しいので，禁じ手とされていましたが，本日ここに解禁したいと思います。

参考コラム　AMAPのアルゴリズム

実際問題として，手作業でマッピングする方も，まずAMAPで自動発生をさせたあとでエディタで手直しするというパターンが多くなると思います。そのためにも，AMAPのアルゴリズムを知っておくべきでしょう。ちょっと複雑ですが，図を見ながら理解してください。

・アトリビュートファイルの変更

マッピングの対象となるアトリビュートの各々に，以下４つのデータが加わる。

```
    mapwind  ( 0 0 255 255 )
    mapview  ( 0 0 255 255 )
    mapsize  ( 0 0 255 255 )
    colormap (<画像ファイル>)
```

〈画像ファイル〉は，AMAP実行時に指定されるファイル名が入るが，mapwind, mapview, mapsizeの各パラメータは固定値となる。

・形状ファイルの変更

貼り付ける画像ファイルは，256×256の固定としている。だから，uv座標は図ａのように決定する。

図a　(0, 0)　(255, 0)　u　(0, 255)　v

またマッピングされるポリゴンが図ｂのとき，AMAPにかける前の形状データは以下のようになっている。

```
art test
prim poly ( 500 100 200 第１頂点
            600  50 300 第２頂点
            650 200 350 第３頂点
          )
```

図b　3Dのポリゴン　第１頂点　第２頂点　第３頂点

このとき，以下のようにして，uvベクトルを求める。この時点では，原点(0, 0)は定まらず，uv座標ではないただのベクトルである点に注意（図ｃ）。

ｕベクトル　：　第２頂点－第１頂点
ｖベクトル　：　第３頂点－第２頂点のベクトルの，ｕベクトルとの直交成分

図c　ｕベクトル　ｖベクトル　1　2　3

このとき，uv平面上でのポリゴンの各頂点の値を求め（図ｄ），
　ｕの最小値，最大値
　ｖの最小値，最大値　を得る。
次に，
　ｕの最小値をｕの0
　ｕの最大値をｕの255
　ｖの最小値をｖの0
　ｖの最大値をｖの255　とし，
このuv座標での各頂点のuv座標を得る（図ｅ）。

この結果得られるマッピングは図ｆのようになる。

この例からわかるように，貼り付けられる画像は，ポリゴンの形によって，縦横に引き伸ばされる。また，第１～第３頂点によって，貼り付けられる方向が決定されるため，CADで形状データを作成する際，どの頂点から作成するかが非常に重要となる。図ｇに，この例と同じポリゴンに対し，頂点の順番が異なった場合の例を記す。

理由は，今回のバージョンアップサービスのために，AMAP（自動マッピングツール）を制作しているからです（“制作したからです”と現在完了形で書けないところが悲しいのですが，そのへんの事情は「各読者通達事項」のコーナーをご覧ください）。

従来マッピングを行うためには，どの画像のどの部分をどの方向にどのように貼り付けるかということを，1つひとつの面ごとに指定してやる必要がありました。AMAPでは，アトリビュートと画像ファイルの一覧が表示されますので，マウスでこのアトリビュートの面にはこの画像を貼り付けると指定するだけで，マッピングに必要な形状ファイル，アトリビュートファイルを生成してくれ……る予定です(よっ弱い)。これさえあれば，誰にでもお手軽にマッピングができるようになる……はずです（まだ試したことがない)。

こんなに便利なAMAPですが，細かな指定がなくなり，すべて自動でやっている以上，微妙な表現ができなくなっているのも事実です。そこで，今回は手作業でマッピングをするのに必要な情報を公開しましょう。まず，画像4（21ページ）をご覧ください。凶悪なゴジラの出現によって，帝都東京が恐怖のどん底に叩き落とされている雰囲気がよく伝わってくるでしょう(ウソ)。この画像（特にゴジラの顔など）は，本来マッピングの使い方とは異なりますが，表現方法としては面白いと思います。私は，今回初めてマッピングを使ってみましたが，この画像の制作がほんの数時間（大部分は2Dの画像データの作成）なのですから，手作業でも意外と実用性があるといえるでしょう。

マッピングを実行するために必要なデータは，アトリビュートファイルと形状ファイルの2つに記述されます。もちろん貼り付ける画像ファイルを用意する必要はありますが，作画時（REND実行時）には，この画像を指定する必要はなく，/Gオプションをつけるだけでマッピングが行われます。

・例　REND GOJIRA.FRM ＊.SUF ＊.ATR /G

まず，アトリビュートファイルに記述されるのは以下の4つのデータです。

```
mapwind ( u1 v1  u2 v2 )
mapview ( u1 v1  u2 v2 )
mapsize  ( u1 v1  u2 v2 )
```

図3　MOYOU.PIC

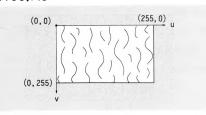



図4　[UVPOLY] GOJIRA.SUF

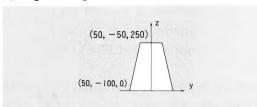
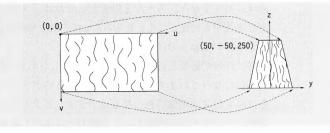



図5　マッピング例

モデラー高津のLOGIN

今月は，マニュアルにも載っていない高度な表現の特集ということで，ネット上で提案されたFFEの応用例を紹介しましょう。

Graphic Galleryのウネウネとしたアニメーションをご覧ください。DōGA内で「気色悪い」「趣味悪い」と絶賛された代物です。このようなサイン曲面や球などの規則的な形状をCADで制作するのはなかなか面倒なのですが，実は，この形状データ，なんとFFEで制作したものなのです。

FFEは従来フレームファイルを制作するツールとして利用してきました。しかし，マニュアルを読んでみましょう。どこにもフレームファイルの記述に関する部分がありません。FFEは，¥～¥ の数式を計算するだけで，それ以外のところには何を書いてもそのまま出力されます。ですから，フレームデータを書けばフレームファイルが出力され，形状データを書けば形状ファイルが出力されるのです。もちろんそれ以外のファイルでも，インタプリタ型のCを使う感覚でなんでもできます。

簡単な例として，九九の表を出力させてみましょう。#repと#endrepの間を，xが1から9まで繰り返されます。¥～¥の部分が計算されて出力されますから，このファイルをFFEにかけるとかけ算の九九の表ができるわけです。

例　九九の表

```
@2.0@
#rep( x, 1, 9 )
¥ 1*x, 2*x, 3*x, 4*x, 5*x, 6*x, 7*x, 8*x, 9*x ¥
#endrep
```

Graphic Galleryのサイン曲面は，中心からの半径に対するコサインをZ値とし，法線ベクトルも計算して，スムースシェーディングがかけられるようになっています。高さを決める関数f（x，y）を変えればいろんな曲面を表現できるでしょう。今回のサイン曲面のリストは，掲載すると大きくなりすぎますので，ご希望の方はネットにアクセスしてください。

さて，今月のアップデータはかまたデザインのビッグウィンドウワゴンです。今ならビーチパラソルとビキニGALもついてきます。

今年の春の「見体験フェア」で，自動車メーカーのデザイナーの方から，“新車のプレゼンテーションに使えるだろうか”という質問を受け，“努力すればできないことはないでしょう”といいかげんに答えました。大阪に戻ってから本当にそんな本格的な使い方が可能だろうかと心配になって，自分で制作してみたのがこのワゴンです。ですから，15秒のCM風のアニメーションになっています。形状デザインも3時間程度ですし，CMの完成度もプレゼンテーション用には十分なデキでそこそこ満足しています。

ところで，このGAL（名はLUNNAと申します）は平面でできていますが，こういった表現もなかなか面白いと思います。制作も楽ですし，へたに3Dで作って不気味なアンドロイドとなるよりはイラストチックな仕上がりになります。皆さんもかわいがってやってください。

colormap (〈画像ファイル名〉)

colormapで貼り付ける画像ファイルを指定するというのは容易に予想できますが，ほかの3つはさっぱりわからないでしょう。ここではとりあえず，ただ単純に256×256ドットの画像を貼り付ける設定として，

```
mapwind ( 0 0  255 255 )
mapview ( 0 0  255 255 )
mapsize  ( 0 0  255 255 )
```

という値を入れておきましょう。まず最初の注意として，これら4つのデータはアトリビュートファイルの中で1度記述すればよいのではなく，RGBの指定と同様に各アトリビュートごとに必要であることを理解してください。

・例　GOJIRA.ATR

```
atr dou { col ( rgb ( 1 1 1 ) )
          amb ( 0.5 )
          dif ( 0.5 )
          mapwind  ( 0 0  255 255 )
          mapview  ( 0 0  255 255 )
          mapsize  ( 0 0  255 255 )
          colormap ( MOYOU.PIC )
        }
```

この例は，ゴジラの胴体部分に，図3のような模様(MOYOU.PIC)を貼り付けようとしています。tra(透過率)やspc(スペキュラー)を省略していますが，特に意味はありません。ちゃんと指定してやれば，マッピング&半透明も可能ですし，もちろんamb，dif，spcの与え方で陰影が変わるのはいつもと同じです。

特に注目が必要なのはcolです。rgbが(1,1,1)つまり真っ白になっていますが，よく考えるとそのポリゴンの色は貼り付ける画像の色になるはずです。ですから，画用紙が白いのと同じ理由で，その面の色は文字どおり白紙の状態にしておくわけです。ここで何か色をつけておくと，色画用紙の上に絵を描くように色が混じってしまいます。正確にいうと，R，G，Bごとにandを取りますので，赤いポリゴンの上に青い画像をマッピングすると，真っ黒になってしまう(1×0=0)わけです。貼り付ける模様を白と黒だけで描いて，赤い面，青い面にマッピングし，赤いゴジラ，青いゴジラを作るという使い方もできます。しかし，赤，青の模様の画像を用意するほうがわかりやすいので，マッピングをするときは，無条件で白にすると覚えておいてまず問題はないでしょう。

次に形状ファイルです。CGAシステムで扱う形状データはポリゴンですが，実はこのポリゴンにも4種類があります。

- poly　：通常のポリゴンデータ
- shade　：スムースシェーディングができるデータ
- uvshade　：マッピングとスムースシェーディングができるデータ
- uvpoly　：マッピングができるデータ

polyは，各項点の3次元座標(x，y，z)の値が羅列しているだけのもっとも基本的なデータです。逆にいうと，poly以外のデータは，REND以外のほとんどのプログラムが対応していないので，たとえばMIRRにかけて，左右反転するなどといったことはできません。

shadeは，CADでスムースシェーディングをかけると生成されるデータです。3次元座標のほかに，各項点での法線ベクトルが必要となりますが，この法線ベクトルをエディタで与えるのは，よほど単純な形状でない限り人間技ではありません。CADのシェーディングの機能はバグがあるままですし，操作性も悪いので使いものになりません。これもAMAPが完成してから使えるデータといえるでしょう。

uvshadeは，スムースシェーディングとマッピングを

各読者通達事項
〈バージョンアップサービス遅れる!〉

CGAコンテストのビデオ配布は予定より早く発送してきたので，“DōGAも発送に慣れてきたんだなぁ”と油断しているユーザーの皆さん，ご安心ください。バージョンアップサービスは，しっかり遅れる予定です。

実は，せっかくのバージョンアップなのだから，夏休みで余裕もあることだし!，2回生の修行も兼ねて，もう少しいろんなツールを制作しようということになったのです。画像データの左右を反転させるというような，非常に簡単なものから，マッピングやスムースシェーディングのデータを簡単に生成するというパワフルなものまで各種制作中です。

以下，制作中のプログラムとプログラマを紹介いたします。もし，バージョンアップのディスクにそのプログラムが入っていなければ，誰が落としたのかひとめでわかるさらし者のコーナーです。

*

AMAP：アヤシゲ守屋……スムースシェーディング，マッピング用データ生成ツール
・難易度No.1　頭を抱えながらも果敢に制作中

REP：教育的寺田……画像データの色を置換したり，透明ビットの操作ができる
・すでに稼動中。なかなか実用性が高いと好評

WIRE：GAVAN島田……ワイヤーフレームアニメーション
・先日バイクで事故り，包帯まきまきにて，落とす候補No.1!

MFONT：岩室……文字データを形状データに変換する
・アルゴリズムもまだ考えていない。これではたして完成するか?

HEDRO：小立……正多面体作成ツール
・楽勝。もう完成したから，マッピング対応検討中

WIPER：中田……シーンのつなぎ目のワイプ効果生成ツール
・結構マンマシンインタフェイスがたいへん。完成危ぶまれる

BW：中西……画像データを白黒に2値化する
・楽勝!　だけどプログラム経験0の初心者。大丈夫だろうか

TURNS：不明……画像データの左右を裏返しにする

SEPIA：不明……回想シーン用に，カラー画像を白黒，セピア調に変換する
・上級生がやれば1日でできる。新入生のだれかにやらせよう

兼ね備えたもので，３次元座標，法線ベクトル，uv座標のすべてが必要となります。しかし，法線ベクトルを求めるのが人間技ではない以上，このデータも手作業で扱うわけにはいきません。

というわけで，uvpolyが今回のメインとなります。通常のpolyのデータと比べてみましょう。

・例［poly］　GOJIRA.SUF

```
atr dou
prim poly  ( 50  -50 250
             50   50 250
             50  100   0
             50 -100   0 )
```

・例［uvpoly］GOJIRA.SUF

```
atr dou
prim uvpoly ( 50  -50 250      0   0
              50   50 250    255   0
              50  100   0    255 255
              50 -100   0      0 255 )
```

まず，prim polyをprim uvpolyに置換します。ひとつの形状データのなかで，polyとuvpolyが混在していても問題ありませんから，マッピングしたいポリゴンだけ行ってください。次に，各項点の座標値ｘ，ｙ，ｚの後ろにuv座標を与えます……。ところでいままではぐらかしていましたがuv座標ってなんでしょうか。

uv座標は，貼り付ける画像の座標です。一般では，3Dの座標はｘ，ｙ，ｚで，2Dの座標はｘ，ｙを使いますが，マッピングのように3Dと2Dの両方を同時に扱う場合，ｘ座標といってもどっちかわからなくなってしまいます。ですから，ｕは貼り付ける画像のｘ座標，ｖはｙ座標のことなのです。

つまり，形状データの各項点にuv座標を与えるというのは，各項点に画像データのどのドットが対応するかを，いちいち指定してやることなのです。上記の例のアトリビュートと形状データで作画させると図５のようになります。このように，貼り付ける画像が長方形で，3Dのポリゴンが台形だった場合，模様はポリゴンの形に応じて引き伸ばされるわけです。

以上がマッピングの基礎です。なんといってもuv座標を与えるのが大変なので，AMAPを使用せず全部手作業で行うのでしたら，ポリゴンは四角形か三角形にとどめておく必要があります。

続マッピング

基礎編で今回は終わろうと思っていたのですが，それではAMAPと大差ないので実用編を続けます。少しずつ難しくなってきますので，がんばってついてきてください。

まず，三角形です。uv座標の設定が問題ですが，ゴジラが襲っている東京タワーの例で解説すると図６のようになります。タワーのてっぺんのuv座標が問題です。

・例　TOWER.SUF

```
obj suf tower {
        atr tower
        prim uvpoly(    0  0 500    127   0
                      200  0   0      0 255
                     -200  0   0    255 255)
}
```

柚姫の明るい悩み相談室

暑い……。ごめんね，いきなり暑苦しいセリフで。姫は夏より冬が好きなもので，暑さに姫の頭はすっかり溶けてます。数学のテストがひどいのも，英語の出席日数が危ういのもみんなこの暑さが……と，つい悪いことは全部夏のせいにしたくなるんですよ。けど，お便りもきていることだしがんばってやろうかなぁ。

（早速お便りをもらってうれしい姫）

＊

Ｑ：１カ月ほど前に，コンテストのビデオのお金を郵便振替で払ったのですが，まだ届きません。もし，このあとまだやることがあるんだったら，教えてください。

姫：最近，CGAシステム，ビデオ，バージョンアップそして追加カンパなどいろいろな名目でDōGAの口座に振り込んでいただいています。ですから，何が目的で振り込んだのかを明記しておかないと，不精な担当者がみんな追加カンパと見なしてしまう恐れがありますのでご注意ください。それから，お名前と連絡先（電話）もお忘れなく。

（名なしの苦情ハガキを手に途方に暮れる姫）

Ｑ：CGAコンテストのビデオのカンパを払ってないんですが，どうすればいいでしょうか。

姫：あくまでカンパということですので，気にすることはありません。もし送ってくれるのなら，締め切りはありませんのでお金と暇のあるときにいつでもどうぞ。カンパはお金でも物でも，もちろん肉体労働でも歓迎します。ちなみにいままでに届いたカンパの一端を披露しますと，サニーレタス１箱，ササニシキ１袋，ハードディスク１コ，なぞの飲料水１ケース……などなど。

（お金のほうがいいぞという会計係を無視する姫）

Ｑ：柚姫とあき姫，どちらがかわいいですか？

姫：世の中，見なきゃよかった，知らなきゃよかったということは多々あるものです。どーしてもというのなら大阪まで会いにきてくれてもいいですけどね。

（こんなお便り載せなきゃよかったと思う姫）

Ｑ：バージョンアップサービス申し込みます。ところでカンパはこのハガキ（カモメール）で当たる賞品というのはいかがでしょう？

姫：バージョンアップサービスで送るディスクにプログラムが入っていたら当たり，入っていなかったら残念ですが，次のチャンスを待っててください。

（しょせん人生はバクチだと思う姫）

Ｑ：DōGAのメインスタッフの方々は京大or阪大生だそうですが，どうすれば京大or阪大に入れるでしょうか（Oh!Xって，受験雑誌でしょう？）。

姫：京大に入るには語学として，英語のほかに京言葉の試験があります。また阪大に入るには実技として漫才が課せられます。がんばって大阪のボケとつっこみを勉強してください。

（実技は満点だった姫）

Ｑ：忙しくてCGAを制作する時間がなかなか取れないのですが，皆さんはどのようにして時間を作られているのでしょうか？

姫：あなたの周りに灰色の服を着た男たちがいませんか。そうです，DōGAでは，活動の合間に「時間銀行」の経営も行っているのです。

（映画版「モモ」の笑顔が好きな姫）

お手紙お待ちしてます！

ちなみにこれを図7のような11角形にすると，途中の点のuv座標を求めるのが，非常に面倒になってしまうわけです。ポリゴンの大きさを，縦横 256 にすると，ポリゴンの3D座標と画像データのuv座標が一致して，簡単に求めることができますが，制限が多くあまり実用的ではありません。

次にゴジラの顔です。この場合，ゴジラの顔の画像ファイルが 256×256 で横長なのに対してマッピングするポリゴンが正方形なので，そのままやると横方向が縮められて，面長でかつ左右に黒の余白ができてしまいます。これを解決する方法は3通りあります。

まずひとつめは，ゴジラの画像を描くとき左右いっぱいまで使った横長にひしゃげた顔にしておく方法です。これだと形状ファイル，アトリビュートファイルを書き換える必要はありませんが，描きにくいのは事実です。

次は，形状ファイルのuv座標を修正する方法です。uv座標は，貼り付けたい画像の位置ですので，画像の隅である必要はないわけです。

・例　GOJIRA.SUF

```
atr kao
prim uvpoly ( 140   60 250    40   0
              140  -60 250   215   0
              140  -60 370   215 255
              140   60 370    40 255 )
```

そして最後はアトリビュートファイルを修正する方法です。

```
mapwind ( u1 v1  u2 v2 )
```

この mapwind は，貼り付ける画像ファイルの有効範囲を設定しています。ですから，

```
mapwind ( 0 0  255 255 )
```

とすれば，256×256の画像ファイルの隅から隅までをマッピングの対象とすることになります。今回の場合，貼り付けたいのは，(40，0)～(215，255)なので，

```
mapwind ( 40 0  215 255 )
```

とします(mapview，mapsize は変更しない)。このような mapwind の使い方は，よくあるでしょう。たとえば，お絵描きソフトで，木目模様を描こうとしたが，256×256の大きさに描くのは面倒で，真ん中辺に小さく描いてしまったというときも，その部分だけをマッピングの対象にすれば問題ありません。ただ，あまり貼り付ける画像データが小さいと，大きなポリゴンにマッピングした際，ドットの粗さが露骨に出てしまう場合があります。

また，貼り付ける画像として，512×512の画像ファイルを使いたい場合（メモリはたくさん消費します），

```
mapwind ( 0 0  511 511 )
```

のように設定しなければいけません。

しかし，mapwind で 512までの範囲を指定しているのに，その画像が256×256の画像ファイルだった場合，「ウィンドウの指定が不正です」とエラーになってしまいます。

図6　TOWER.SUF

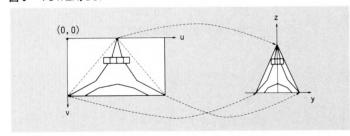



図7　11角形の場合

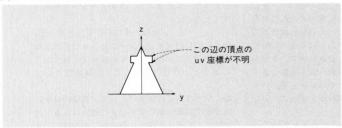



図8　ゴジラの顔（基本）

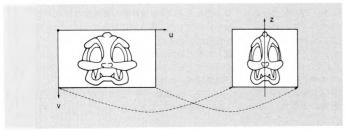



図9　ひしゃげさせた場合

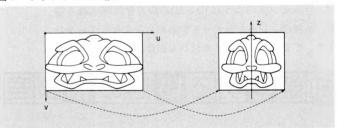



図10　uv座標を修正する場合

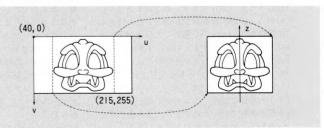
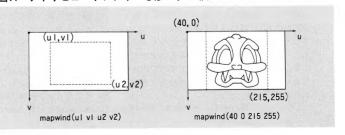



図11　アトリビュートファイルを修正する場合

さらに，応用編として次のような特殊効果もできます。いままでの例では，貼り付ける画像がゆがんだりしないように，矛盾のないuvデータを与えていました。そこで今度は，わざと変なデータを与えましょう。

・例　BUIL.SUF

```
    obj suf buil  {
            atr  buil
            prim  uvpoly （ 0  －100  300       0    0
                             0    100  300     255    0
                             0    100    0     255    0
                             0  －100    0       0  255 ）
         }
```

この例では，ポリゴンの２つの頂点が同一のuv座標を持っています。その結果，画像ファイルの一部が引き伸ばされて，ゆがんでマッピングされます。この性質を利用すると，ゴジラの放射能火炎によって半分溶かされたビルを表現できます。

さらに，対応するuv座標がねじれるようにマッピングすると，もう予想もできないような絵になります。

・例　TEST.SUF

```
        prim  uvpoly （ 0  －250  500       0    0
                        0    250  500     255  255
                        0    250    0     255    0
                        0  －250    0       0  255 ）
```

こんなねじ曲がったマッピングの仕方，どのようなときに役に立つかといえば，たぶんなんの役にも立たないでしょう。面白いから紹介してみただけです……。

さて，あんまりいいかげんなことを書いていると怒られそうなので，実用性のある使い方を紹介しましょう。アトリビュートのデータは標準どおり，（ 0 0 255 255 ）のとき，形状データのuv座標を255より大きな値にしてみます。

・例　TEST.SUF

```
        prim  uvpoly （    0  －250  500       0    0
                           0    250  500     511    0
                           0    250    0     511  511
                           0  －250    0       0  511 ）
```

この例では縦横２倍の値にしました。すると，貼り付ける画像の模様が縦横２個ずつ並んだ絵ができます。511を400にすると，縦横に１個半並んだ絵になります。

つまり，uv座標というのは，０～255 である必要性はまったくありません。uv座標は，貼り付ける画像が，縦横に無限に繰り返されているものなのです。

たとえば，大理石の模様を貼り付ける場合，小さなポリゴンと大きなポリゴンの両方に同じマッピングをしていては，きめ細かさが異なって，不自然になってしまいます。大きなポリゴンは，それに比例してuv座標を大きくしてやれば，大理石の模様が連続して，きめ細かさを一定にすることができます。ただこの場合，連続しても模様の境目が目立たないような画像ファイルをつくってやる必要があります。

以上が形状データのuv座標の応用です。それでは最後に，アトリビュートの中で無視されていたmapview，mapsizeについて解説しましょう。まず，

　　mapview （ u1 v1   u2 v2 ）

ですが，これは気にしないでください。本来はちゃんとした意味があるのですが，RENDでは手を抜いて無視しています。（ u1 v1   u2 v2 ）には，とりあえずmapsizeと

図12　マッピングのための木目模様の例

図13　ゆがみを利用した場合（その１）

図14　ゆがみを利用した場合（その２）

図15　マッピング利用例

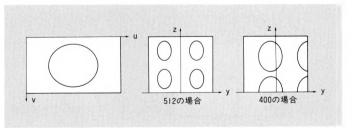
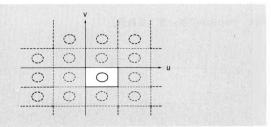



図16　図15の応用

同じ値を入れてください。ほかの値を入れてもなんの変化もありませんが，mapview自体を省略すると，偉そうに「マッピングデータがおかしい」と文句をいってきます。

次に，

mapsize (u1 v1 u2 v2)

ですが，これはuv座標の目盛りの付け方を設定しています。すでに解説した mapwind は貼り付ける画像の有効範囲を指定していますが，この時点ではuv座標が設定された訳ではありません。この mapsize によって，貼り付ける画像の左上隅のuv座標と，右下隅のuv座標を指定することによって，uv座標全体が定義されるのです。

・例　TEST.ATR

```
    mapsize ( 100 200  300 400 )
TEST.SUF
    prim uvpoly (  0 -250 500      0   0
                   0  250 500    256   0
                   0  250   0    256 256
                   0 -250   0      0 256 )
```

この例では，貼り付ける画像の左上隅が（100,200），右下が(300,400)という座標になり，uv座標の原点(0,0)の位置も大きく変わってしまいます。uv座標系が変わるのですから，形状データのuv座標もそれに応じた値にする点にご注意ください。

このmapsizeの使い方は2通りあります。まずひとつめは，uv座標を正規座標系で記述することです。

・例　TEST.ATR

```
    mapsize ( 0 0  1 1 )
TEST.SUF
    prim uvpoly (  0 -250 500      0   0
                   0  250 500      1   0
                   0  250   0      1   1
                   0 -250   0      0   1 )
```

図17　mapsizeとmapwindの関係

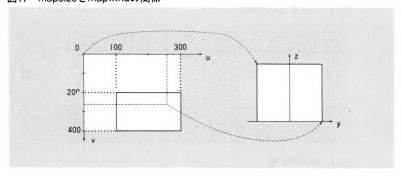
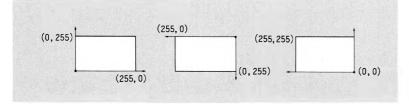



図18　mapsizeの使い方（反転例）

このようにすると，uv座標を生成するプログラムを制作する場合など楽になるケースが多く，また，上記のように，形状データのuv座標が美しく記述できます。

もうひとつの使い方は，mapsizeの値を変えるだけで，上下左右を反転することです。

・例　上下反転

```
    mapsize ( 0 255  255 0 )
左右反転
    mapsize ( 255 0  0 255 )
上下左右反転
    mapsize ( 255 255  0 0 )
```

以上がマッピングに関するテクニックです。今回用いた貼り付け用の画像は，ゴジラの顔からテストパターンまで，すべて Z'sSTAFF で描きました(Z'sSTAFFの画像データのままでは使用できません。バージョンアップサービスに付いてくるツールによって，CGAシステムの画像データに変換しています)。マッピングするときかなりゆがみますので，きっちりした絵より少しぼやけたような絵のほうが仕上がりはきれいです。文字を描いたような画像データを貼り付けても，ほとんど読めません。Z'sSTAFF で描いた絵だけでなく，CGAシステムの3Dの画像をマッピングすることで，面白い効果が出せるような気もしますので，皆さんもいろいろやってみてください。

おわりに

アンケートを集計しておりますと，“もっと具体的に”とか，“図を用いて解説してほしい”というご意見がかなりあることに気がつきました。そこで今回は，できるだけ図を用意し具体的にファイルのリストも掲載するようにしました。少しはわかりやすくなったとは思いますが，紙面の都合上，本文の量が減り，内容的には物足りないような気がします（というより，マッピングの解説に手間取ってしまった)。表現を増すテクニックは，まだまだありますので，今回掲載できなかった分については，また次の機会に回したいと思います。皆さんも，何か面白いアイデアがありましたら，ぜひお便りください。

今回公開されたテクニックはあくまで道具ですので，目的になってしまわないようにご注意ください。意味もなく，映り込みや空気遠近法を用いたカットを作っていても，作品にはなりません。空気遠近法は現在制作中の作品の中で，グランドキャニオンの壮大感を出すために設けられた機能です。これらのテクニックは，あくまでも作品制作のなかで生かしてください。

さて，このところ隔月連載となっていますが，来月は今月大阪で開かれましたあるイベントのレポートを少しだけ行う予定です。そして再来月では，やっぱり使えない人のために，MAX田口君が「宇宙要塞CADの逆襲」をお送りします。お楽しみに。

特集2　2Dグラフィック続論

ADVANCED 2D GRAPHICS

デジタルペインティングへの道

本格的なCGには1600万色が必要？　アナログRGB画像が持つ可能性を過小評価してはいないか？　これまであまりにも古い概念に縛られていたような気はしないか？　オーバーサンプリングと桒野式アルゴリズムが導く2Dグラフィックの世界を見よ。2Dの可能性をさらに追ってみよう。

Tan Akihiko 丹 明彦

先月掲載したアンチエリアシング付き描画関数は，標準のグラフィック関数に比べれば大きな可能性を持ってはいるが，なんといっても使いにくい。あのままでは，機能はともかく，操作性は従来のグラフィック関数から一歩も進化していない。

サンプルで載っけた女の子の（ような）顔は僕が作ったものだ。あんな使いにくいものを誰かに使わせるような仕打ちはさすがにできなかった。で，方眼紙に下絵を描いてぽちぽちと座標を打ち込んでいったのである。おお，これではまるで人間デジタイザではないか。まったく最低であった。そしてできあがった絵はどうかというとご覧のとおり。よくいえばシンプル，率直にいえば描き込みが足りない絵。もちろん根気が続かなかったせいである。僕の画才などたかがしれているものだが，それでも下絵のほうがまだましな絵だった（と自分では思っている）。

しかし，この方式のいいところは，いったん輪郭のデータを作ってしまえばあとは極楽という点に尽きるであろう。髪の毛をどんな色にしようと，スクリーントーンにしようと自由。髪の毛を少しずらしてスクリーントーンにして貼れば影のような効果も出せる。わずかな時間でアンチエリアスのかかった絵を何枚でも作れる。手を変え品を変えして，いろんな絵を堪能できた。それに比べて前半の作業は地獄地獄。

そこで思った。面倒なことは計算機にやらせてしまえと。わがX68000にはマウスという強力なポインティングデバイスが標準装備されているではないか。座標をとるのさえ簡単になれば，もう少しましな絵が描けるかもしれない。さらにはもっとたくさんの人にも使ってもらえるかもしれないと。

これが点列エディタ

さて今回は

今回は面倒な人間デジタイズの作業を計算機に肩代わりさせることをたくらむことにする。マウスで点をポイントすれば，あとはプログラムのほうで必要なデータを計算してくれる。要するにグラフィックエディタだ。

といっても，見慣れたグラフィックエディタとは少し違う。どちらかといえばドロー系のツールだ。点をつないでいって絵を作る。途中の点をちょっとずらしたり，ひと通り描いたあとに線の太さを変えたりといったことが自在にできるわけだ。

最後にBASICの描画プログラムを出すようにしておけば，色を変えてみたり，塗りつぶしてみたり，スクリーントーンにしてみたりといった楽しい作業だけが待っている。いってみれば作業の重い部分は計算機に押しつけて，おいしい部分だけを人間様がかっさらおうという魂胆なのである。名づけて点列エディタ。

使い方の前に作り方

前回に味をしめ，X-BASICの外部関数を作ることにした。ただ描画関数が前よりも少ないので，今回も採用した分割コンパイルが面倒な作業に感じられるかもしれない。

点列エディタにはテキスト画面に線を引くための関数を使う。線分および点列をテキスト画面に描く関数で，前回のanti.fncとペアでBASICに組み込んで使う。その名もtline.fnc。

コンパイルの要領は前回と同じである。環境も前回と同じで構わない。以下駆け足で説明する。次のリストをテキストエディタで打ち込む。

- tline.s　（外部関数ヘッダ）
- anti.h　（マクロ定義ファイル）
- main.c　（引数リスト宣言）
- tlines.c　（点列描画）

これらのうち，

- anti.h　（マクロ定義ファイル）
- main.c　（引数リスト宣言）

の2つは前回のものと同じである。

打ち込んだら，それぞれをコンパイルおよびアセンブルする。つまり，

```
as /u ～.s
cc /L ～.c
gcc –c ～.c
```

のようにする。詳しくは前回の説明をご覧いただきたい。

最後にリンクフェイズ。

```
lk /o tline.fnc tline.o main.o tlines.o %lib%¥clib.a (%lib %¥gnulib.a) %lib%baslib.a
```

あとはX-BASICに組み込んで起動すればよい。まずBASICディレクトリに，いま作った外部関数を転送する。

```
copy tline.fnc a:basic2
```

それからBASICディレクトリ上のコンフィギュレーションファイル(標準ではbasic.cnf)を書き換える。たとえば，

```
FREE   = 128
WIDTH  = 64
BEEP   = ON
CAPS   = OFF
FUNC   = GRAPH
FUNC   = MOUSE
FUNC   = PIC
FUNC   = ANTI
FUNC   = TLINE
```

のようにする。今月の点列エディタには，マウス関係の関数を使っているので，mouse.fncも忘れずに組み込んでおくこと。

なお，打ち込んでちゃんと動くようにな

ったらファイル入出力の前後にERROR ON/OFFを加えておいてほしい。

先月のフォロー

ここでちょっとした拡張をしておこう。先月の自由曲線のプログラム（pts_curve.c）中に，単位ベクトル化する関数normalize( )があったが，それに次のように1行付け加えてコンパイルし直しておくことをすすめる。別にバグというわけではないのだが，この変更をしておくと，点列エディタでちょっとした技が使えるようになる。それはあとのお楽しみ。

```
void normalize(v1,v2)
vector v1,v2;
{
  double l;
  l=length(v1) ;
  if (l<1.0e-10) l=1.0; ←この1行
  v2[0]=v1[0]/l;
  v2[0]=v1[0]/l;
  return;
}
```

これは，表面的には零ベクトルの処理をきちんとするというだけの意味しか持たないが，実は曲線の表現力を若干(かなり？)上げる効果があるのである。それはおいおい説明することになるであろう。

点列エディタ使用上の注意

さてBASICが立ち上がったら，さっそく点列エディタを打ち込んで走らせよう。BASICのリストとしてはやや長めだが，行番号を使いそうな命令は避けてあるので，行番号抜きのリストを，使い慣れたテキストエディタで入力してからload@命令で読ませてもいい。

キーボードとマウス併用である。キー操作は表1に示す。はっきりいって，機能も表に書いてあることそのままである。操作についてはそのうちに慣れてくるだろうから，ここでは，特に気をつけるべきことやちょっとしたテクニックを書いておこう。

・マウスを動かすと，ラバーバンドを引っ張りはするが，それだけでは点の座標は確定しない。たとえば，この時点でなにかキーを押すとラバーバンドはキャンセルされる。その点は，マウスの左ボタンをクリックして初めて確定する。確定した点と未確定の点（ラバーバンド上の点）は色が違うのですぐにわかる。

・点列の終点を編集しているときに左クリックすると，すぐに次の点を作ってラバーバンドに乗せる。対して，点列の途中の点を編集しているときに左クリックすると，その点の位置が変わるだけで新しい点はできない。こうしたほうが自然な操作感覚が得られるであろう。点列の途中に新しく点を作りたい場合はi (insert-point) キーを使う。

・どうしても角（かど）のある曲線を描きたい場合が出てきた場合の処置は次のようにする。1カ所に点を2度打つのである。プログラム上，曲線はここで一度切れたようなかたちになり，したがって角ができる。点の重ね打ちをするためには，点列の終点であれば左クリックを2回することになるだろうし，そうでなければiキーを使うことになるだろう。iキーは，その場に点のコピーをひとつ作る働きをするので，1回押すだけでいいはず。またここでうっかりマウスを動かしてしまっても，左クリックするまでは確定しないので，あわてずにp (previous-point)やn(next-point)などのキーで逃げるとよい。

・さらに応用技として，直線と曲線をひとつの点列に同居させることもできる。重ね打ちした点を2組作り，隣接させる。すると，そのあいだの線は必ず直線になる。これを使えばサーキットのコース図だって簡単に描ける(？)。

・描画コマンドには，カレントの点列だけ

図1　点の重ね打ち

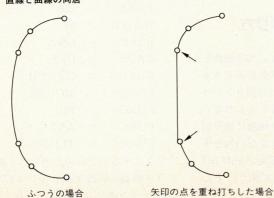



表1　点列エディタのコマンド

コマンド	機能
(カーソル移動)	
p　previous-point	同じ点列内の前の点に移る
n　next-point	同じ点列内の次の点に移る
P　previous-pts	前の点列に移る
N　next-pts	次の点列に移る
(点列のパラメータ操作)	
t　thinner-point	カーソルの指す点を細くする
f　fatter-point	カーソルの指す点を太くする
T　thinner-pts	現在の点列全体を細くする
F　fatter-pts	現在の点列全体を太くする
c　toggle-curve	現在の点列の自由曲線モードを切り替える
C　toggle-cyclic	現在の点列の循環モードを切り替える
(点および点列の追加・削除)	
a　append-point	現在の点列にひとつ点を付け加える
A　append-pts	新しい点列を作る（点列名を入力すること）
i　insert-point	カーソル位置の点を複製する
k　kill-point	カーソル位置の点を削除する
K　kill-pts	現在の点列を削除する
(描画)	
d　draw-current-pts	現在の点列を描く
D　draw-all-pts	全点列を描く
(ファイル入出力)	それぞれファイル名を入力すること
w　write-pts-file	点列ファイルをセーブする
r　read-pts-file	点列ファイルをロードする
o　output-basic-program	BASICプログラムを出力する
s　save-pic-file	画面をPICファイルにセーブする
l　load-pic-file	PICファイルを画面にロードする
(終了)	
q　quit	画面をクリアして終了する
e　end	画面をクリアせずに終了する

を描くd(draw-current-pts)キーと全点列を描くD(draw-all-pts)キーがあるが，あまり描画速度は速くない。特に後者は，線の数が多くなってくるとだんだんユウウツになってくることだろう。だから，適当なところでs(save-pic-file)キーやl(load-pic-file)キーを使って，PICファイルをアンドゥバッファの代わりに使う。こうすればいちいち全部描き直す必要もない。
・現在やっている作業の続きを次回も続けたいのなら，必ずw(write-pts-file)キーで保存し，次回はr(read-pts-file)キーで復元すること。o(output-basic-file)キーで出力するBASICのプログラムは，実行可能にはなっているが（行番号がつかないのでload＠で読み込むことが必要），点列エディタに再び読み込むことはできない。

*

実は，冒頭で自由曲線のルーチンに拡張を行ったのは，点の重ね打ちを許すためだったのである。これにより角のある曲線や，曲線と直線の同居が可能になる。点が重ね打ちしてあるということは，そこで零ベクトルが発生することを意味する。先月のバージョンでは，零ベクトルを許していなかったのだ，というより，おかしな動作をしてしまう。別にバグではなく，それが仕様だったのだ（しつこいぞ）。

絵を描くスタイルは各人の自由である。気に入った線が描けるたびにPICファイルにセーブする人，PICファイルをアンドゥ用バッファとして使う人，PTSファイルをまめに保存する人，BASファイルに落としてから本格的に加工する人，どんなスタイルをとってもいいと思う。

自由課題として，バージョンアップを考えてもいいだろう。たとえば点列を拡大・縮小・移動・回転するというのは，あるととても便利であろう。基本的には，点および点列を削除/追加しないなら，特別な操作はいらない。座標を変えるだけだったらそれほど困難を伴わないと思われる。ほかにもペイントくらいは組み込むべきだとか，改良の余地はいくらでもある。ツリー構造を導入するのは相当な大仕事になるかもしれない。あると便利だろうけど。

*

どうもX-BASICには,「もうひといきのところだったのにねえ」というところが多々ある。それが点列エディタのリストにもちらほら反映している。いちばん困ったのは，再三指摘されていることだが，せっかくローカル変数を文法的に許しているのに，配列を引数としてとる関数が書けないことである。おかげで配列をグローバル変数にとって転送するような泥縄的解決法を使うはめになった。ほかにもprint usingが表示するデータをひとつしかとれないだとか，配列にした文字列はlinputやfreadsと相性が悪いとか，いろいろある。

それから，このプログラムはコンパイルできない。ついでにいうと，点列エディタでoコマンドを使って生成するBASICプログラムもコンパイルできない。理由のひとつは，前回と今回で作った外部関数と同じ機能を持ったライブラリを用意していないこと，もうひとつは2次元配列への代入を多用していること，などである。

この点列エディタの操作性は決してほめられたものではない。反応もよくない。それにもかかわらずコンパイルしようと考えないのは，速度の点で足をいちばん引っ張っているのが実は描画関数そのもの（aa_lines（）関数）だからなんだよねえ。うーんお粗末。

今月は2本立てだ

さてここで趣向を変えて，「遊べる2Dグラフィック」に手をつけてみよう。

ひとつは65536色のハンデをはね返す，マッハバンドフリーの高品位グラデーション。もうひとつは偶然性と意外性を楽しむ，ランダムフラクタルによる自然物ジェネレータ。気楽に使っていただきたい。

これまたX-BASICの外部関数である。ただ，ここまでの内容とはまったく関わりがないので独立して使える。要するにanti.fncを持っていなくても使える。外部関数の名前はmap.fncとした。

それでは作り方を。例によってテキストエディタで以下のファイルを打ち込む。

・map.s　　　（外部関数ヘッダ）
・map.h　　　（マクロ定義ファイル）
・main.c　　　（引数リスト宣言）
・grad_box.c （高品位グラデーション）
・crush.c　　（ランダムフラクタル）

これらのうち，

・main.c　　　（引数リスト宣言）

は前回のものと同じである。

あとは同様。それぞれをコンパイルおよびアセンブルする。

```
as /u ～.s
cc /L ～.c
gcc -c ～.c
```

リンクフェイズも同じ。

```
lk /o map.fnc map.o main.o grad_box.o crush.o %lib%¥clib.a(%lib%¥gnulib.a)%lib%baslib.a
```

X-BASICに組み込んで起動する。

```
copy map.fnc a:¥basic2¥
```

コンフィギュレーションファイルは，たとえば，

```
FREE   =  128
WIDTH  =  64
BEEP   =  ON
CAPS   =  OFF
FUNC   =  GRAPH
FUNC   =  PIC
FUNC   =  MAP
```

のようにする。

アルゴリズムの心

まずグラデーション。

例によって従来のものをこきおろすところから入る（ああ悪い性格)。2Dグラフィックツールの最高峰であるZ'sSTAFFは，最高峰であるがゆえにいろいろと苦情も多くなる。先月はアンチエリアシング対応でないことを攻撃したが，今月はマッハバンド無策をあげつらっていじめてしまうのである（つくづく悪い性格)。

Z'sSTAFFに触ってみた方はご存じだろうが，結構遊べてしまう機能のひとつにグラデーションがある。Z'sSTAFFを使ってみてなかなかやるなと思ったのは，道具を選ばずグラデーションがかけられるところである。ボックスフィルは当然のこと，ペン，ペイント，スキャンコンバージョン，などほとんどすべての道具の色指定にグラデーションが使えるのだ。グラデを使いまくった絵を描いた経験は，誰でも持っているのではなかろうか。

グラデーションは，手軽な操作でそこそこの質感が出せるのでなかなかに重宝である。しかし使いすぎると単に見苦しくなるだけなので，そのうちだんだん使わなくなっていくようである。

さて，グラデーションは確かに便利だが，使ってみるとどうも神経に障るところがあった。それがマッハバンド。あの見苦しい縞々である。以前にも何度かいってきたことではあるが（こればっかり)，少ない階調数でもって滑らかに変化する色を表現しようとしてぶち当たる壁がこのマッハバンドなのである。まったく人間様の目というものはよくできているもので，ちょっとした輝度の変化でも見事に拾い出す。普段は不足を感じない65536色もこのときばかりは不満の種である。

先月のエリアシングは，解像度の不足か

ら生じる不自然さで，今月のマッハバンドは階調の不足から生じる不自然さ。それでは階調が十分にあればマッハバンドは消えるというのだろうか。これが消えるのである。たとえば業務用のCGシステムだと，RGBそれぞれ8ビット（256階調），合計24ビット（約1670万色）が標準であるが，これくらいあれば十分といわれている。事実このレベルになれば支障はまったくない。しかしX68000はRGB各5ビット(32階調)，合計15ビット（32768色），輝度ビットをどううまく使っても16ビット（65536色）である。これを真っ正直に使ったのでは，とてもマッハバンドは解消できない。

それではどうする。先月は，解像度の不足から生じたエリアシングを，多階調表現によるアンチエリアシングで補った。そこでものの道理として，今月は階調の不足から生じたマッハバンドは解像度でもって補ってやることにしよう。

といってもその正体は，最近とみに出番の多い桒野式アルゴリズムの応用（オリジナルは65536色を2色に落とすものだったが，これはフルカラーを32768色に落とすように改造してある）である。色を滑らかに変化させようとして失敗するのだから，細かな揺らぎを持たせてやれば目をだますこともできるであろう。桒野式アルゴリズムの説明については，もうすっかり有名だろうからパス。

グラデーションを高画質にするだけではあまりに芸がないので，斜めグラデーションもつけてみた。斜めグラデーションはZ'sSTAFFにはなかった機能だが，最近見たPDSのグラフィックツールに搭載されていたので，さっそくいただくことにした。ただいただくだけでは面白くないので，桒野式アルゴリズムを使って品質の高いものを作ったというわけ。まあ速度が少し遅いのはご愛敬。斜めグラデーションについては，わりと真面目に色を計算している（だから遅い)。点と直線の距離を求める公式を使い，開始色と終了色のあいだで補間する。この補間した色の精度を高めにとっておき，画面に出力する段階で桒野式アルゴリズムを使って階調の不足を補う。

＊

次はランダムフラクタル。

フラクタル図形はご存じであろう。自己相似構造を持った図形を総称してこう呼ぶ。フラクタル図形は再帰呼び出しを使って生成するのが定石である。たとえば樹木の生成には，まず幹を作り，その先の枝には少し細い木を，さらにその先の枝にはもう少し細い木を……，という具合に次々と下請けのルーチンを呼び出し，より小さな木を作らせる。そして最後にできあがる図形は木によく似ている。このあたりのことは少し前のC言語の特集で僕が取りあげている話題だから，少しは参考になるかもしれない。

そこでランダムフラクタルであるが，その名のとおり乱数を使って生成するので，厳密にいえば自己相似性のある図形とはいいがたい。しかし長い目で見れば（もう少し難しくいえば統計学的には）自己相似性が見受けられる，ということで，これもフラクタルの仲間に入れる。やはり再帰呼び出しを使って生成するのが常道である。

もっとも簡単なランダムフラクタルは1次元のランダムフラクタルである。まず始点と終点を決める。その2点間を結ぶ線分をぐしゃぐしゃに曲げるというのがもっとも直感的な説明だろう。具体的には，まず線分をその中点で2つに分ける。そしてその中点を適当にずらす。この中点の変位を乱数で決める。こうしてできた2本の線分を，それぞれさらに2分割して中点にランダムな変位を加えて……，と，あとはこの繰り返し。できあがる図形は，まるで稲妻のようになるであろう。

これをちょっと拡張したのが2次元のランダムフラクタル。今回使うのもこれである。まず1枚の平面を用意する。それを縦

図2　斜めグラデーションの描き方

・横に2分割する（4枚の小さな平面ができる）。すると格子点が5個新たにできる。この5個の格子点の高さを，やはり乱数を使って変えてやる。あとは同じことで，4枚の小平面に対して再帰分割を繰り返していけばよい。こうしてできる図形は，よく山の表現のモデルとして使われるが，自然な山に見せるためには，上の説明だけではやや不十分である。格子点の高さを変える際に，

1)　その格子点と隣り合った格子点の高さの平均をまずとる。そのあとに乱数で変位を加える。

2)　再帰が深くなると分割も細かくなる。だから変位の振幅もそれにしたがって小さくしてやる。

といった工夫が必要であろう。

さて，2次元のランダムフラクタルは山の表現に向いているといったが，今回はそれとは違った利用のしかたをしよう。自然界にはフラクタルの構造が多く見られるといわれる。そこで，計算機にフラクタルを導入することで自然物を擬似的に表現しようという試みがあちらこちらでなされている。ランダムフラクタルもそのひとつなのである。

ここに1枚の絵があるとしよう。これを，ランダムフラクタルを使って変形する。たったこれだけのことなのだが，かなり面白い効果が出るので驚きだ。絵をどう変形するのか。各格子点の変位（つまり山の高さ）を，歪みの度合いと考えるのである。山が高いほど歪みも大きい。山の起伏が激しいほど絵の乱れ方も派手になる。

山の高さや起伏の様子は，パラメータとしてBASIC側から好きに与えられるようにした。どんな変形が起こるのか，実行してみるまでわからないが，偶然と意外性を楽しむのも風流なものである。

さて使い方

くどい説明はさっさと終わって，おいしいところを味わうほうに進むことにする。

＊

グラデーションからいこう。斜めかつ高画質のグラデーションだが，残念ながら矩形領域（つまりボックスフィル）のみの対応である。先月制作した描画関数のタイルパターンの代わりに組み込むことは十分可能なのだが，それには大幅な改造が必要そうだったのであきらめた。それよりも単独で使うことのほうがメリットが多そうだ。

使い方は至って簡単。

grad__box(x1, y1, x2, y2, gx, gy, cs, ce)

(x1, y1)−(x2, y2):矩形領域

(gx, gy):色が変化する方向のベクトル

cs, ce:開始色と終了色

ランダムフラクタルの“雲”

2～3回試せば，使い方はすぐにわかることと思う。少しデキる人なら，マウスで方向と領域を指定してグラデーションを描かせるプログラムも簡単に書けることだろう。

＊

そしてランダムフラクタル。

使い方は2段階に分かれる。

1)　ランダムフラクタル格子を生成する。

random__fractal(L0;float, β;float, seed)

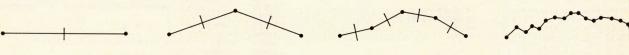
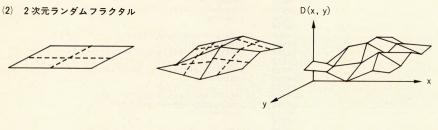
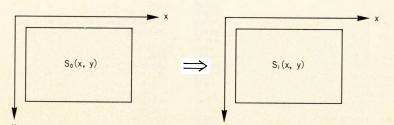
L0:初めの振幅

β:振幅の減衰係数

seed:乱数系列の初期値

L0とβは実数で指定する。L0は歪みの度合いの目安で100～200が標準的な値であろう。βは0.1～0.5がいい線だろうか。この辺は，経験がものをいうし，第一いちば

図3　ランダムフラクタル

(1)　1次元ランダムフラクタル

(2)　2次元ランダムフラクタル

画像S_0を，ランダムフラクタル格子D(x, y)を用いて画像S_1に変形する。

$S_1(x, y)=S_0(x, y+D(x, y))$

今回はランダムフラクタル格子の大きさを128×128としたので，それより大きな画像を変形する際にモザイク模様が出るのを避けるために補間を行っている。

ん効果的な値など，僕にもわからない。そこもランダムのよさなのだ(？)。とりあえずは前記の値で始めて，いろいろと値を変えて試すのが無難な線であろう。

2)　ランダムフラクタル格子を使ってグラフィック画面を変形する。

crush(x1, y1, x2, y2, k;float)
(x1,y1)−(x2,y2):矩形領域
k:歪みの倍率

kは実数で指定する。実はこの値ができあがりをいちばん大きく左右する。だから，ランダムフラクタル格子を生成するのは起動時の1回だけにしておいて，画像を生成するたびに違うkの値を指定すればそれで十分なような気がする。なにぶん乱数相手なので，なにが起こっても不思議はない。

＊

ここでおすすめの使い方があるので，紹介しよう。さきほどのグラデーションとコンビにして使うのである。

1)　ランダムフラクタル格子を生成する。

random＿fractal (200.0, 0.1, 12345)

2)　青白のグラデーションを描く。

grad＿box (0, 0, 255, 255, 2, −5, rgb (0, 0, 31), rgb (31, 31, 31))

3)　画像を変形する。

crush (0, 0, 255, 255, 0.5)

もっとも重要なパラメータは3)のkの値(この例では0.5) である。これをいろいろ変えて効果を試したい場合は，毎回フレッシュなグラデーションを用意すべきだろう。すなわち2)～3)を繰り返す必要があるのだ。ここで3)だけ繰り返しても，画像がどんどん崩れていくだけだ (それはそれで面白いが)。

で，実行結果だが，雲のような絵が出てきたのではないかと思う。元絵はグラデーションだから，かなり規則的な絵のはずなのに，一度ランダムフラクタル処理をかけてやるだけで，結構面白い効果が出せる。2)のグラデーションをもっと複雑なパターンにしてみたり，色を変えてみたりして，いろいろなテクスチャの表現に挑戦してみてもらいたい。

元絵やパラメータしだいでは，炎，大理石，なども作れそうである。レイトレーシングのテクスチャマップにそのまま使ってもいいくらいの画質は保証する。応用範囲は広い。処理が遅いのが玉にキズといえようか (もっともこれは僕が高速化をさぼったせいだが)。

終わりに

今回はネタが散漫になってしまった。点列エディタは先月のフォローみたいなものだが，あとの2つは，処理速度をちゃんとしたうえで，今後出てくるグラフィックツールにも組み込んでほしいくらいの機能である。この分野ではPC-9801シリーズに出ている2次元グラフィックツールに一日の長がある。とりわけ，自然物の表現というか，表情豊かな2次元グラフィックを作るための努力を惜しんでいないツールがいくつかあり，充実した環境といえる。

X68000ならそれらと同等以上のものが作れても少しも不思議はないのだが，こういうアプローチをとったグラフィックツールは不思議と見かけない。Z'sSTAFFが他の追随を許さないのなら，それはそれでよい。別の方面から超えればよいだけのこと。そんなツールの出現を願ってやまないのである。

参考文献

ランダムフラクタルを用いた木目と大理石表現技術，横井茂樹・岡田稔，PIXEL，1988.8，pp.87−92

リスト1

```
10000 /* 点列エディタ
10010 key 1,"console 0,31,1@M"
10020 /*
10030 str crlf,eof
10040 crlf=chr$(&HD)+chr$(&HA)
10050 eof=chr$(&H1A)
10060 dim str flag(1)={" LINES "," CURVE "}
10070 dim str type(1)={"NORMAL ","CYCLIC "}
10080 dim str pts_name(31)     /* 各点列の名前
10090 dim char pts_flag(31)    /* 曲線として描画するか
10100 dim int pts(31,99,2)     /* 点列
10110 dim int tmp1(99,2),tmp2(1999,2)
10120 dim int pts_cur(31)
10130 dim int ptslist(31,1)
10140 dim int freelist(31)
10150 int first,last,fre
10160 int i,j,k
10170 char c
10180 /* 点列リストの初期化
10190 first=-1:last=-1:fre=0
10200 for i=0 to 31
10210   freelist(i)=i+1
10220   ptslist(i,0)=-1
10230   ptslist(i,1)=-1
10240 next
10250 freelist(31)=-1
10260 i=0:j=1   /* i番目の点列のj番目の点を処理する
10270 /* 画面などの初期化
10280 screen 1,3,1,1
10290 mouse(1):mouse(4)
10300 msarea(0,0,511,511)
10310 setmspos(256,256)
10320 open_minibuffer()
10330 /* メインループ
10340 while 1
10350   c=set_point(i,j)
10360   /* レベル3：点列がなくてもよいコマンド
10370   if (c='q') or (c='e') then break
10380   if (c=12) then wipe():continue    /* 'CLR'キー
10390   if (c='s') then save_pic():continue
10400   if (c='l') then load_pic():continue
10410   if (c='r') then {
10420     wipe()
10430     close_minibuffer():cls:open_minibuffer()
10440     read_pts()
10450     i=first:j=pts_cur(i)
10460     setmspos(pts(i,j,0),pts(i,j,1))
10470     continue
10480   }
10490   if (c='A') then {
10500     if first>-1 then tdraw(i,1,0)
10510     k=append_pts()
10520     if k>-1 then i=k:j=append_point(i):pts_cur(i)=j
10530     continue
10540   }
10550   /* レベル2：点列が1本以上必要なコマンド
10560   if (first=-1) then {
10570     cls:print "There is no pts. Try 'A' command.";
10580     pause()
10590     continue
10600   }
10610   if (c='N') and (ptslist(i,0)>-1) then {
10620     tdraw(i,1,0)
10630     i=ptslist(i,0):j=pts_cur(i)
10640     tdraw(i,1,1)
10650     setmspos(pts(i,j,0),pts(i,j,1))
10660     continue
10670   }
10680   if (c='P') and (ptslist(i,1)>-1) then {
10690     tdraw(i,1,0)
10700     i=ptslist(i,1):j=pts_cur(i)
10710     tdraw(i,1,1)
10720     setmspos(pts(i,j,0),pts(i,j,1))
10730     continue
10740   }
10750   if (c=0) and (pts(i,0,0)=1) then {
10760     j=append_point(i):pts_cur(i)=j
10770     continue
10780   }
10790   if (c='a') then {
10800     k=append_point(i)
10810     if k>-1 then j=k:pts_cur(i)=j
10820     continue
10830   }
10840   if (c='K') then {
10850     tdraw(i,1,0)
10860     k=delete_pts(i)
10870     if (first>-1) and (k>-1) then {
10880       i=k
10890       j=pts_cur(i)
10900       setmspos(pts(i,j,0),pts(i,j,1))
10910       tdraw(i,1,1)
10920     }
10930     continue
10940   }
10950   if (c='c') then {
10960     pts_flag(i)=1-pts_flag(i)
10970     continue
```

```
10980   }
10990   if (c='C') then {
11000     tdraw(i,1,0)
11010     pts(i,0,1)=1-pts(i,0,1)
11020     tdraw(i,1,1)
11030     continue
11040   }
11050   if (c='D') then {
11060     wipe()
11070     k=first
11080     while 1
11090       if k=-1 then break
11100       draw(k)
11110       k=ptslist(k,0)
11120     endwhile
11130     continue
11140   }
11150   if (c='w') then write_pts():continue
11160   if (c='o') then output_program():continue
11170   /* レベル1：点列が1本以上、点も1個以上必要なコマンド
11180   if (pts(i,0,0)<2) then {
11190     cls
11200     print "There is no point. Try 'a' command or press lef
t-button.";
11210     pause()
11220     continue
11230   }
11240   if (c=0) then {
11250     if j=pts(i,0,0) then {
11260       k=append_point(i)
11270       if k>-1 then j=k:pts_cur(i)=j
11280     }
11290     continue
11300   }
11310   if (c='n') and (j<pts(i,0,0)) then {
11320     j=j+1:pts_cur(i)=j
11330     setmspos(pts(i,j,0),pts(i,j,1))
11340     continue
11350   }
11360   if (c='p') and (j>1) then {
11370     j=j-1:pts_cur(i)=j
11380     setmspos(pts(i,j,0),pts(i,j,1))
11390     continue
11400   }
11410   if (c='t') and (pts(i,j,2)>0) then {
11420     pts(i,j,2)=pts(i,j,2)-1
11430     continue
11440   }
11450   if (c='f') then pts(i,j,2)=pts(i,j,2)+1:continue
11460   if (c='T') then {
11470     for k=1 to pts(i,0,0)
11480       if pts(i,k,2)>0 then pts(i,k,2)=pts(i,k,2)-1
11490     next
11500     continue
11510   }
11520   if (c='F') then {
11530     for k=1 to pts(i,0,0)
11540       pts(i,k,2)=pts(i,k,2)+1
11550     next
11560     continue
11570   }
11580   if (c='i') then {
11590     k=insert_point(i,j)
11600     if k>-1 then j=k:pts_cur(i)=j
11610     continue
11620   }
11630   if (c='k') then {
11640     tdraw(i,1,0)
11650     k=delete_point(i,j)
11660     if k>-1 then j=k:pts_cur(i)=j:tdraw(i,1,1)
11670     continue
11680   }
11690   if (c='d') then draw(i):continue
11700 endwhile
11710 /* 終了
11720 close_minibuffer():cls
11730 if c='q' then wipe()  /* 'e'のときは画面を消さずに終了
11740 mouse(0):mouse(2)
11750 end
11760 /* 点列の情報を表示する
11770 func print_status(i,j,x,y)
11780   if first=-1 then return()
11790   cls    /* pts(i,0,0) 番目の点はラバーバンド上（未確定）
11800   print pts_name(i);flag(pts_flag(i));type(pts(i,0,1));
11810   print j-1;"/";pts(i,0,0)-1;"(";x;",";y;") ";pts(i,j,2);
11820 endfunc
11830 /* エラー発生時などの入力待ち
11840 func pause()
11850   int rx,ry,bl,br
11860   str s
11870   beep
11880   while 1
11890     msstat(rx,ry,bl,br)
11900     s=inkey$(0)
11910     if rx or ry or bl or br or (s<>"") then break
11920   endwhile
11930 endfunc
11940 /* ラバーバンドの表示
11950 func rubber_band(i,j,x,y,p,m)
11960   int n,t
11970   if first=-1 then return()
11980   n=pts(i,0,0)       /* 点列の頂点数
11990   if n<2 then return()
12000   t=pts(i,0,1)       /* 点列のタイプ
12010   if j=1 then {      /* 始点
12020     tline( x,y,pts(i,j+1,0),pts(i,j+1,1),p,m )
12030     if t=1 then {    /* CYCLIC な場合
12040       tline( pts(i,n,0),pts(i,n,1),x,y,p,m )
12050     }
12060     return()
12070   }
12080   if j=n then {      /* 終点
12090     tline( pts(i,j-1,0),pts(i,j-1,1),x,y,p,m )
12100     if t=1 then {    /* CYCLIC な場合
12110       tline( x,y,pts(i,1,0),pts(i,1,1),p,m )
12120     }
12130     return()
12140   }
12150   tline( pts(i,j-1,0),pts(i,j-1,1),x,y,p,m )
12160   tline( x,y,pts(i,j+1,0),pts(i,j+1,1),p,m )
12170   return()
12180 endfunc
12190 /* マウス左ボタンかキー入力を得る
12200 func char set_point(i,j)
12210   int x,y,rx,ry,bl,br
12220   str s
12230   mspos(x,y)
12240   print_status(i,j,x,y)
12250   while 1
12260     rubber_band(i,j,x,y,0,0)
12270     s=inkey$(0)
12280     if s<>"" then break
12290     mspos(x,y)
12300     rubber_band(i,j,x,y,0,1)
12310     msstat(rx,ry,bl,br)
12320     if (rx<>0) or (ry<>0) then print_status(i,j,x,y)
12330     if bl<>0 then {
12340       if first=-1 then break
12350       rubber_band(i,j,x,y,0,0)
12360       rubber_band(i,j,pts(i,j,0),pts(i,j,1),1,0)
12370       pts(i,j,0)=x:pts(i,j,1)=y
12380       rubber_band(i,j,pts(i,j,0),pts(i,j,1),1,1)
12390       if (pts(i,0,1)=1) and (pts(i,0,0)=3) then tdraw(i,1,
1)
12400       while (bl<>0):msstat(rx,ry,bl,br):endwhile
12410       if (rx<>0) or (ry<>0) then print_status(i,j,x,y)
12420       break
12430     }
12440   endwhile
12450   return( asc(s) )  /* マウスのボタンが押された場合は 0 を返す
12460 endfunc
12470 /* draw-current-pts：i番目の点列の描画
12480 func draw(i)
12490   int j
12500   for j=0 to pts(i,0,0)-1
12510     tmp1(j,0)=pts(i,j,0)
12520     tmp1(j,1)=pts(i,j,1)
12530     tmp1(j,2)=pts(i,j,2)
12540   next
12550   tmp1(0,0)=tmp1(0,0)-1
12560   pts_oversample(tmp1)
12570   if (pts_flag(i)=1) then {
12580     pts_curve(tmp1,0,0,tmp2)
12590     aa_lines(tmp2,65534)
12600   } else aa_lines(tmp1,65534)
12610 endfunc
12620 /* draw-all-pts：全点列の描画
12630 func tdraw(i,p,m)
12640   int n,j
12650   n=pts(i,0,0)-1:if n=0 then return()
12660   for j=0 to n
12670     tmp1(j,0)=pts(i,j,0)
12680     tmp1(j,1)=pts(i,j,1)
12690     tmp1(j,2)=pts(i,j,2)
12700   next
12710   tmp1(0,0)=tmp1(0,0)-1
12720   pts_oversample(tmp1)
12730   tlines(tmp1,p,m)
12740 endfunc
12750 /* append-point：点を追加する
12760 func int append_point(i)
12770   int n,x,y
12780   n=pts(i,0,0)+1
12790   if n=100 then {
12800     cls
12810     print "append-point: no new point is available.";
12820     pause()
12830     return(-1)
12840   }
12850   pts(i,0,0)=n
12860   if n=1 then {
12870     mspos(x,y)
12880     pts(i,n,0)=x
12890     pts(i,n,1)=y
12900     pts(i,n,2)=16
12910   } else {
12920     pts(i,n,0)=pts(i,n-1,0)
12930     pts(i,n,1)=pts(i,n-1,1)
12940     pts(i,n,2)=pts(i,n-1,2)
12950   }
12960   return(n)
12970 endfunc
12980 /* insert-point：点を挿入する
12990 func int insert_point(i,j)
```

```
13000   int n,x,y
13010   n=pts(i,0,0)
13020   if n=99 then {
13030     cls
13040     print "insert-point: no new point is available.";
13050     pause()
13060     return(-1)
13070   }
13080   n=n+1
13090   pts(i,0,0)=n
13100   if n=2 then {
13110     mspos(x,y)
13120     pts(i,n,0)=x
13130     pts(i,n,1)=y
13140     pts(i,n,2)=16
13150   } else {
13160     while 1
13170       pts(i,n,0)=pts(i,n-1,0)
13180       pts(i,n,1)=pts(i,n-1,1)
13190       pts(i,n,2)=pts(i,n-1,2)
13200       n=n-1
13210       if n=j then break
13220     endwhile
13230   }
13240   setmspos( pts(i,j,0),pts(i,j,1) )
13250   return(j)
13260 endfunc
13270 /* delete-point:点を削除する
13280 func int delete_point(i,j)
13290   int n,x,y,k
13300   n=pts(i,0,0)
13310   if n=1 then {
13320     cls:print "delete-point: no point exists.";
13330     pause()
13340     return(-1)
13350   }
13360   for k=j to n-1
13370     pts(i,k,0)=pts(i,k+1,0)
13380     pts(i,k,1)=pts(i,k+1,1)
13390     pts(i,k,2)=pts(i,k+1,2)
13400   next
13410   n=n-1
13420   pts(i,0,0)=n
13430   if j>n then j=n
13440   setmspos(pts(i,j,0),pts(i,j,1))
13450   return(j)
13460 endfunc
13470 /* append-pts:点列を追加する
13480 func int append_pts()
13490   int fw
13500   str stmp
13510   if fre=-1 then { /* これ以上点列を作れない場合
13520     cls:print "new-pts: no new pts is available.";
13530     pause()
13540     return(-1)
13550   }
13560   if first=-1 then { /* 点列リストが空の場合、新規に作る
13570     first=fre:last=fre
13580     fre=freelist(fre)
13590     ptslist(last,0)=-1
13600     ptslist(first,1)=-1
13610   } else {             /* 点列リストに新しい点列を追加する
13620     fw=fre
13630     fre=freelist(fre)
13640     ptslist(last,0)=fw
13650     ptslist(fw,0)=-1
13660     ptslist(fw,1)=last
13670     last=fw
13680   }
13690   cls:linput "new-pts: ";stmp
13700   pts_name(last)=stmp
13710   pts_flag(last)=1  /* デフォルトは曲線とする
13720   pts(last,0,0)=0   /* まだ点列は空
13730   pts(last,0,1)=0   /* CYCLIC でない
13740   pts(last,0,2)=0   /* オーバーサンプリングなし
13750   return( last )
13760 endfunc
13770 /* delete-pts:点列を削除する
13780 func int delete_pts(i)
13790   int fw, bk
13800   if first=-1 then {
13810     cls:print "delete-pts: no pts exists.";
13820     pause()
13830     return(-1)
13840   }
13850   fw=ptslist(i,0)
13860   bk=ptslist(i,1)
13870   freelist(i)=fre
13880   fre=i
13890   if (fw=-1) and (bk=-1) then {
13900     first=-1
13910     last=-1
13920     return(fre)
13930   }
13940   if (fw=-1) then {
13950     last=bk
13960     ptslist(bk,0)=-1
13970     return(bk)
13980   }
13990   if (bk=-1) then {
14000     first=fw
14010     ptslist(fw,1)=-1
14020     return(fw)
14030   }
14040   ptslist(bk,0)=fw
14050   ptslist(fw,1)=bk
14060   return(fw)
14070 endfunc
14080 /* メッセージ表示領域を作る
14090 func open_minibuffer()
14100   console 30,1,1
14110   cls
14120 endfunc
14130 /* メッセージ表示領域を閉じる
14140 func close_minibuffer()
14150   cls
14160   console 0,31,1
14170 endfunc
14180 /* ファイル名に拡張子をつける
14190 func str put_extend( filename;str, ext;str )
14200   int i
14210   i = instr( 1, filename, "." )
14220   if i>0 then filename = left$( filename, i-1 )
14230   return( filename+ext )
14240 }
14250 /* write-pts-file:点列データをセーブする
14260 func write_pts()
14270   str filename
14280   int fp,i,j
14290   dim int tmp(299)
14300   cls:linput "write-pts-file: ";filename
14310   filename = put_extend( filename, ".PTS" )
14320   fp = fopen( filename, "c" )
14330   cls:print using "writing pts-file '@'...";filename;
14340   i=first
14350   while i>-1
14360     fwrites( pts_name(i)+crlf, fp )
14370     fputc( pts_flag(i), fp )
14380     tmp(0) = pts(i,0,0)-1
14390     tmp(1) = pts(i,0,1)
14400     tmp(2) = pts(i,0,2)
14410     for j=1 to pts(i,0,0)-1
14420       tmp(j*3  ) = pts(i,j,0)
14430       tmp(j*3+1) = pts(i,j,1)
14440       tmp(j*3+2) = pts(i,j,2)
14450     next
14460     fwrite( tmp, 3+(pts(i,0,0)-1)*3, fp )
14470     i=ptslist(i,0)
14480   endwhile
14490   fclose( fp )
14500   print "done";
14510 endfunc
14520 /* read-pts-file:点列データをロードする
14530 func read_pts()
14540   str filename,stmp
14550   int fp,i,j
14560   dim int tmp(299)
14570   cls:linput "read-pts-file: ";filename
14580   filename = put_extend( filename, ".PTS" )
14590   fp = fopen( filename, "r" )
14600   cls
14610   if fp=-1 then {
14620     print using "no such file as '@'.";filename;
14630     return()
14640   }
14650   print using "reading pts-file '@'...";filename;
14660   i=0:first=i
14670   while ( feof(fp)=0 )
14680     freads( stmp, fp )
14690     pts_name(i)=stmp
14700     pts_flag(i)=fgetc(fp)
14710     fread( tmp, 3, fp )
14720     pts(i,0,0) = tmp(0)
14730     pts(i,0,1) = tmp(1)
14740     pts(i,0,2) = tmp(2)
14750     fread( tmp, pts(i,0,0)*3, fp )
14760     for j=1 to pts(i,0,0)
14770       pts(i,j,0) = tmp(j*3-3)
14780       pts(i,j,1) = tmp(j*3-2)
14790       pts(i,j,2) = tmp(j*3-1)
14800     next
14810     j=append_point(i):if j>-1 then pts_cur(i)=j
14820     ptslist(i,0)=i+1
14830     ptslist(i,1)=i-1
14840     tdraw(i,1,1)
14850     i=i+1
14860   endwhile
14870   last=i-1
14880   ptslist(first,1)=-1
14890   ptslist(last,0)=-1
14900   fclose( fp )
14910   print "done";
14920 endfunc
14930 /* 点列の1単位の組を文字列にする
14940 func str pts_st(i,j)
14950   str stmp
14960   stmp = itoa(pts(i,j,0))+","+itoa(pts(i,j,1))+","+itoa(pt
s(i,j,2))
14970   return( stmp )
14980 endfunc
14990 /* output-basic-program:BASIC プログラムを出力する
15000 func output_program()
15010   str filename
15020   int fp,i,j
```

▶今月で100号かぁ。うちのOh!X (MZ) も90冊になりました。記念に専用の本棚を買ってやろうと思ってます。
谷口充弘(17)東京都

```
15030   cls:linput "output-basic-program: ";filename
15040   filename = put_extend( filename, ".BAS" )
15050   fp = fopen( filename, "c" )
15060   cls:print using "writing basic-program '@'...";filename;
15070   fwrites( "/* "+filename+crlf, fp )
15080   fwrites( "screen 1,3,1,1"+crlf, fp )
15090   fwrites( "dim int tmp(99,2)"+crlf, fp )
15100   i=first
15110   while i>-1
15120     fwrites( "dim int "+pts_name(i)+"(1999,2)"+crlf, fp )
15130     fwrites( "tmp={", fp )
15140     pts(i,0,0)=pts(i,0,0)-1
15150     fwrites( pts_st(i,0), fp )
15160     fwrites( crlf, fp )
15170     for j=1 to pts(i,0,0)
15180       fwrites( "    ,", fp )
15190       fwrites( pts_st(i,j), fp )
15200       if j=pts(i,0,0) then fwrites( "}", fp )
15210       fwrites( crlf, fp )
15220     next
15230     pts(i,0,0)=pts(i,0,0)+1
15240     fwrites( "pts_oversample( tmp )"+crlf, fp )
15250     if pts_flag(i)=1 then {
15260       fwrites( "pts_curve( tmp,0,0,"+pts_name(i)+" )"+crlf
, fp )
15270     } else {
15280       fwrites( "pts_move( tmp,0,0,"+pts_name(i)+" )"+crlf,
 fp )
15290     }
15300     fwrites( "aa_lines( "+pts_name(i)+",rgb(31,31,31) )"+c
rlf, fp )
15310     i=ptslist(i,0)
15320   endwhile
15330   fwrites( "end"+crlf, fp )
15340   fwrites( eof, fp )
15350   fclose( fp )
15360   print "done";
15370 endfunc
15380 /* save-pic-file:グラフィック画面をセーブする
15390 func save_pic()
15400   str filename
15410   cls:linput "save-pic-file: ";filename
15420   filename = put_extend( filename, ".PIC" )
15430   cls:print using "saving pic-file '@'...";filename;
15440   pic_save( filename,0,0,511,511 )
15450   print "done";
15460 endfunc
15470 /* load-pic-file:グラフィック画面をロードする
15480 func load_pic()
15490   str filename
15500   int fp
15510   cls:linput "load-pic-file: ";filename
15520   filename = put_extend( filename, ".PIC" )
15530   fp = fopen( filename, "r" )
15540   if fp=-1 then {
15550     cls:print using "no such file as '@'.";filename;
15560     pause()
15570     return()
15580   }
15590   fclose( fp )
15600   cls:print using "loading pic-file '@'...";filename;
15610   pic_load( filename,0,0 )
15620   print "done";
15630 endfunc
```

リスト2

```
==================  tlines.c  ==================
  1: /****** テキスト画面の点列描画 ******/
  2:
  3: #include  "anti.h"
  4:
  5: #define   TXVRAM  0xE00000
  6: #define   PLSIZE  0x020000
  7: #define   RASTER  0x80
  8:
  9: unsigned char OVERSAMPLE_NOTYET[]="オーバーサンプリング座標に変換してください";
 10: unsigned char BAD_PLANE[]="プレーン番号には 0~3 を指定してください";
 11:
 12: void  c_tline( x1, y1, x2, y2, plane, mode )  /* 1区間だけ(線分1本)描く
*/
 13: int x1, y1, x2, y2; /* 座標 */
 14: int plane;          /* テキスト VRAM のプレーン番号 */
 15: int mode;           /* 描画モード( 0:reset, 1:set, 2:xor ) */
 16: {
 17:   int             dx2, dy2, sx, sy, i, e, ssp;
 18:   unsigned short  *address, pat;
 19:
 20:   dx2 = ABS( x2-x1 );
 21:   dy2 = ABS( y2-y1 );
 22:   sx  = SGN( x2-x1 );
 23:   sy  = SGN( y2-y1 )*(RASTER/sizeof(short));
 24:
 25:   ssp = B_SUPER( 0 );
 26:   address = (short *)( TXVRAM+plane*PLSIZE+y1*RASTER+(x1/16)*2 );
 27:   pat = ( 0x8000>>(x1%16) );
 28:
 29:   if ( dx2>=dy2 ) {
 30:     e = -dx2;
 31:     i = dx2;
 32:     dx2 *= 2;
 33:     dy2 *= 2;
 34:     for ( ; i>=0; i-- ) {
 35:       if ( mode<2 ) *address |= pat;    /* set   ... OR */
 36:       if ( !(mode&1) ) *address ^= pat; /* reset ... OR+XOR */
 37:       e += dy2;
 38:       if ( e>=0 ) {
 39:         address += sy;
 40:         e -= dx2;
 41:       }
 42:       if (sx>0) {
 43:         pat >>= 1;
 44:         if ( pat==0 ) {
 45:           pat = 0x8000;
 46:           address++;
 47:         }
 48:       } else {
 49:         pat <<= 1;
 50:         if ( pat==0 ) {
 51:           pat = 0x0001;
 52:           address--;
 53:         }
 54:       }
 55:     }
 56:   } else {
 57:     e = -dy2;
 58:     i = dy2;
 59:     dx2 *= 2;
 60:     dy2 *= 2;
 61:     for ( ; i>=0; i-- ) {
 62:       if ( mode<2 ) *address |= pat;    /* set   ... OR */
 63:       if ( !(mode&1) ) *address ^= pat; /* reset ... OR+XOR */
 64:       e += dx2;
 65:       if ( e>=0 ) {
 66:         if (sx>0) {
 67:           pat >>= 1;
 68:           if ( pat==0 ) {
 69:             pat = 0x8000;
 70:             address++;
 71:           }
 72:         } else {
 73:           pat <<= 1;
 74:           if ( pat==0 ) {
 75:             pat = 0x0001;
 76:             address--;
 77:           }
 78:         }
 79:         e -= dy2;
 80:       }
 81:       address += sy;
 82:     }
 83:   }
 84:   B_SUPER( ssp );
 85:
 86:   return;
 87: }
 88:
 89: FUNC  tline( dummy )
 90: DUMMY dummy;
 91: /* int x1, y1, x2, y2, plane, mode */
 92: {
 93:   unsigned int  x1, x2, y1, y2, plane, mode;
 94:
 95:   ARGSET( dummy );
 96:   x1    =IVALUE(1);
 97:   y1    =IVALUE(2);
 98:   x2    =IVALUE(3);
 99:   y2    =IVALUE(4);
100:   plane=IVALUE(5);
101:   mode =IVALUE(6);
102:
103:   if ( plane>3 ) {
104: #ifdef  __GNUC__
105:     asm ( " lea.l _BAD_PLANE,a1" );
106: #else
107: #asm
108:   lea.l _BAD_PLANE,a1
109: #endasm
110: #endif
111:     return ( 1 );
112:   }
113:   c_tline( x1, y1, x2, y2, plane, mode );
114:
115:   return ( 0 );
116: }
117:
118: FUNC  tlines( dummy ) /* 関数本体、全点列を描画する */
119: DUMMY dummy;
120: /* PTS *pts; int plane, mode */
121: {
122:   PTS           *pts;
123:   unsigned int  x1, x2, y1, y2, plane, mode;
124:   int           i, n;
125:
126:   ARGSET( dummy );
127:   ARYSET(1);
128:   pts   =PARYTOP(1);
129:   plane=IVALUE(2);
130:   mode =IVALUE(3);
```

▶編集部の移転先が高輪とはこりゃオドロキ！　ということは泉岳寺の駅で編集部の人とすれちがっているのかなー。
木村亮(17)東京都

```
131:
132:   if ( plane>3 ) {
133: #ifdef  __GNUC__
134:     asm ( " lea.l _BAD_PLANE,a1" );
135: #else
136: #asm
137:   lea.l _BAD_PLANE,a1
138: #endasm
139: #endif
140:     return ( 1 );
141:   }
142:   n=pts[0][0];
143:   if ( pts[0][2]!=OVERSAMPLE ) {
144: #ifdef  __GNUC__
145:     asm ( " lea.l _OVERSAMPLE_NOTYET,a1" );
146: #else
147: #asm
148:   lea.l _OVERSAMPLE_NOTYET,a1
149: #endasm
150: #endif
151:     return ( 1 );
152:   }
153:   for ( i=1; i<n; i++ ) {
154:     x1 = PIX( pts[i][0] );
155:     y1 = PIX( pts[i][1] );
156:     x2 = PIX( pts[i+1][0] );
157:     y2 = PIX( pts[i+1][1] );
158:     c_tline( x1, y1, x2, y2, plane, mode );
159:   }
160:   if ( pts[0][1]==CYCLIC ) {/*点列が循環している場合、終点と始点をつなぐ*/
161:     x1 = PIX( pts[n][0] );
162:     y1 = PIX( pts[n][1] );
163:     x2 = PIX( pts[1][0] );
164:     y2 = PIX( pts[1][1] );
165:     c_tline( x1, y1, x2, y2, plane, mode );
166:   }
167:   return ( 0 );
168: }
```

リスト3

```
================  tline.s  ==================
  1: ****** 外部関数ヘッダ ******
  2:
  3: * tlines.c
  4: *        tline( int x1, y1, x2, y2, plane, mode )
  5: *        tlines( PTS *pts, int plane, mode )
  6:
  7: * インフォメーション・テーブル
  8:
  9:         dc.l    X_INIT
 10:         dc.l    X_RUN
 11:         dc.l    X_END
 12:         dc.l    X_SYS
 13:         dc.l    X_BRK
 14:         dc.l    X_CTRL_D
 15:         dc.l    X_RES1
 16:         dc.l    X_RES2
 17:         dc.l    PTR_TOKEN
 18:         dc.l    PTR_PARAM
 19:         dc.l    PTR_EXEC
 20:         dc.l    0,0,0,0,0
 21:
 22: X_INIT:
 23: X_RUN:
 24: X_END:
 25: X_SYS:
 26: X_BRK:
 27: X_CTRL_D:
 28: X_RES1:
 29: X_RES2:
 30:         rts
 31:
 32: * 関数名テーブル
 33:
 34: PTR_TOKEN:
 35:         dc.b    'tline',0
 36:         dc.b    'tlines',0
 37:         dc.b    0
 38:         .even
 39:
 40: * パラメータ・テーブルへのポインタ
 41:
 42: PTR_PARAM:
 43:         dc.l    TLINE_PAR
 44:         dc.l    TLINES_PAR
 45:
 46: * パラメータ・テーブル
 47:
 48: int_val:        equ $0002 * int
 49: PTS_ary:        equ $0052 * 1D-array of PTS ( 2D-array of int )
 50: void_ret:       equ $ffff * void
 51:
 52: TLINE_PAR:
 53:         dc.w    int_val
 54:         dc.w    int_val
 55:         dc.w    int_val
 56:         dc.w    int_val
 57:         dc.w    int_val
 58:         dc.w    int_val
 59:         dc.w    void_ret
 60: TLINES_PAR:
 61:         dc.w    PTS_ary
 62:         dc.w    int_val
 63:         dc.w    int_val
 64:         dc.w    void_ret
 65:
 66: * 関数へのポインタ
 67:
 68: PTR_EXEC:
 69:         dc.l    _tline
 70:         dc.l    _tlines
 71:
 72:         .even
```

リスト4

```
================  map.h  ==================
  1: /****** 汎用マクロなどの定義 ******/
  2:
  3: #define N_PIXEL   512 /* スクリーンのサイズは 512×512 ピクセル */
  4:
  5: /* X-BASIC からの引数をアクセスする */
  6:
  7: typedef int FUNC;   /* X-BASIC の外部関数:戻り値はエラーコード */
  8: typedef int DUMMY;  /* Cとの引数の受け渡しの違いを吸収するダミー引数 */
  9:
 10: extern unsigned short *par;         /* 一時的な引数リスト */
 11: extern unsigned short *ary[10+1]; /* 一時的な配列リスト:X-BASIC の引数は最大 10
個 */
 12:
 13: #define ARGSET( A ) { par=(unsigned short *)(&A); }     /* 引数 */
 14: #define ARGC          ( par[0] )                        /* 引数の個数 */
 15: #define ATOP( I )     ( (I)*5-4 )                       /* 第 I 引数の先頭 */
 16: #define TYPE( I )     ( par[ATOP(I)] )                  /* 引数の型 */
 17: #define IVALUE( I ) ( *(int *)(&(par[ATOP(I)+3])) )     /* int の値 */
 18: #define CVALUE( I ) ( par[ATOP(I)+4] )                  /* char の値 */
 19: #define FVALUE( I ) ( *(double *)(&(par[ATOP(I)+1])) ) /* float の値 */
 20:
 21: /* その他、便利なマクロ */
 22:
 23: #define ABS( X )  (((X)>0)?(X):(-(X)))          /* X の絶対値 */
 24: #define SGN( X )  (((X)>0)?1:(((X)<0)?(-1):0)) /* X の符号(正負または零)*/
 25:
 26: #define MIN( X, Y ) (((X)>(Y))?(Y):(X)) /* X,Y のっち大きくないほう */
 27: #define MAX( X, Y ) (((X)>(Y))?(X):(Y)) /* X,Y のうち小さくないほう */
 28:
 29: /* R,G,B ごとの輝度を得るためのマスクとビットシフト */
 30:
 31: #define VMASK_B   62    /* 0b0000000000111110 */
 32: #define VMASK_R   1984  /* 0b0000011111000000 */
 33: #define VMASK_G   63488 /* 0b1111100000000000 */
 34:
 35: #define SHIFT_B   1
 36: #define SHIFT_R   6
 37: #define SHIFT_G   11
 38:
 39: /* R,G,B からカラーコードを計算する */
 40:
 41: #define RGB(R,G,B)  ((B)<<SHIFT_B|(R)<<SHIFT_R|(G)<<SHIFT_G)
 42:
 43: /* カラーコードから R,G,B 成分を取り出す */
 44:
 45: #define BLUE(C)   ( ((C)&VMASK_B) >> SHIFT_B )
 46: #define RED(C)    ( ((C)&VMASK_R) >> SHIFT_R )
 47: #define GREEN(C)  ( ((C)&VMASK_G) >> SHIFT_G )
 48:
 49: #define IMAX  31  /* R,G,B の輝度の最大値 */
```

リスト5

```
================  map.s  ==================
  1: ****** 外部関数ヘッダ ******
  2:
  3: * grad_box.c
  4: *        grad_box( int x1, y1, x2, y2, gx, gy, cs, ce )
  5: * crush.c
  6: *        random_fractal( float l0, b [, int seed [, d00, d01, d10, d11
]] )
  7: *        crush( int x1, y1, x2, y2, float k )
  8:
  9: * インフォメーション・テーブル
 10:
```


▶日本橋の某店ではMZ-2521が19,800円で，MZ-80Bが1980円で売られていました。時の流れは早い。
木下達也(18)兵庫県

```
11:         dc.l    X_INIT
12:         dc.l    X_RUN
13:         dc.l    X_END
14:         dc.l    X_SYS
15:         dc.l    X_BRK
16:         dc.l    X_CTRL_D
17:         dc.l    X_RES1
18:         dc.l    X_RES2
19:         dc.l    PTR_TOKEN
20:         dc.l    PTR_PARAM
21:         dc.l    PTR_EXEC
22:         dc.l    0,0,0,0,0
23:
24: X_INIT:
25: X_RUN:
26: X_END:
27: X_SYS:
28: X_BRK:
29: X_CTRL_D:
30: X_RES1:
31: X_RES2:
32:         rts
33:
34: * 関数名テーブル
35:
36: PTR_TOKEN:
37:         dc.b    'grad_box',0
38:         dc.b    'random_fractal',0
39:         dc.b    'crush',0
40:         dc.b    0
41:         .even
42:
43: * パラメータ・テーブルへのポインタ
44:
45: PTR_PARAM:
46:         dc.l    GRAD_BOX_PAR
47:         dc.l    RANDOM_FRACTAL_PAR
48:         dc.l    CRUSH_PAR
49:
50: * パラメータ・テーブル
51:
52: float_val:      equ $0001       * float (引数)
53: int_val:        equ $0002       * int (引数)
54: float_val_omt:  equ $0081       * float (省略可)
55: int_val_omt:    equ $0082       * int (省略可)
56: void_ret:       equ $ffff       * void (戻り値)
57:
58: GRAD_BOX_PAR:
59:         dc.w    int_val
60:         dc.w    int_val
61:         dc.w    int_val
62:         dc.w    int_val
63:         dc.w    int_val
64:         dc.w    int_val
65:         dc.w    int_val
66:         dc.w    int_val
67:         dc.w    void_ret
68:
69: RANDOM_FRACTAL_PAR:
70:         dc.w    float_val
71:         dc.w    float_val
72:         dc.w    int_val_omt
73:         dc.w    int_val_omt
74:         dc.w    int_val_omt
75:         dc.w    int_val_omt
76:         dc.w    int_val_omt
77:         dc.w    void_ret
78:
79: CRUSH_PAR:
80:         dc.w    int_val
81:         dc.w    int_val
82:         dc.w    int_val
83:         dc.w    int_val
84:         dc.w    float_val
85:         dc.w    void_ret
86:
87: * 関数へのポインタ
88:
89: PTR_EXEC:
90:         dc.l    _grad_box
91:         dc.l    _random_fractal
92:         dc.l    _crush
93:
94:         .even
```

リスト6

```
================  grad_box.c  ==================
  1: /****** 斜めグラデーション(マッハバンドなし)******/
  2:
  3: #include  <graph.h>
  4: #include  "map.h"
  5:
  6: #define SC  256 /* 色の精度を確保するための倍率 */
  7:
  8: unsigned char OUTOF_SCREEN[]="指定した座標がスクリーンの範囲外です";
  9:
 10: FUNC  grad_box( dummy )
 11: DUMMY dummy;
 12: /* int x1, y1, x2, y2, gx, gy, cs, ce */
 13: {
 14:   int x1, y1, x2, y2, gx, gy, cs, ce;
 15:   int d0, d, w;
 16:   int tmp, x, y, xs, ys, xe, ye, dx, dy, yy;
 17:   static int  xx[ N_PIXEL ];
 18:   unsigned int  r, g, b, rs, gs, bs, re, ge, be;
 19:   unsigned int  lc, lb, c;
 20:   unsigned int  cg, cr, cb, dg, dr, db;
 21:   static unsigned int bg[2][ N_PIXEL ];
 22:   static unsigned int br[2][ N_PIXEL ];
 23:   static unsigned int bb[2][ N_PIXEL ];
 24:   static unsigned short slbuf[ N_PIXEL ];
 25:
 26:   ARGSET( dummy );
 27:   x1=IVALUE(1);
 28:   y1=IVALUE(2);
 29:   x2=IVALUE(3);
 30:   y2=IVALUE(4);
 31:   gx=IVALUE(5);
 32:   gy=IVALUE(6);
 33:   cs=IVALUE(7);
 34:   ce=IVALUE(8);
 35:
 36:   if ( x2<x1 ) {
 37:     tmp=x1;
 38:     x1=x2;
 39:     x2=tmp;
 40:   }
 41:   if ( y2<y1 ) {
 42:     tmp=y1;
 43:     y1=y2;
 44:     y2=tmp;
 45:   }
 46:
 47:   if ( x1<0 || x2>=N_PIXEL || y1<0 || y2>=N_PIXEL ) {
 48: #ifdef  __GNUC__
 49:     asm( "  lea.l _OUTOF_SCREEN,a1" );
 50: #else
 51: #asm
 52:   lea.l _OUTOF_SCREEN,a1
 53: #endasm
 54: #endif
 55:     return ( 1 );
 56:   }
 57:
 58:   dx = x2-x1;
 59:   dy = y2-y1;
 60:
 61:   if ( gx>=0 ) {
 62:     xs=0;
 63:     xe=dx;
 64:   } else {
 65:     xs=dx;
 66:     xe=0;
 67:   }
 68:
 69:   if ( gy>=0 ) {
 70:     ys=0;
 71:     ye=dy;
 72:   } else {
 73:     ys=dy;
 74:     ye=0;
 75:   }
 76:
 77:   rs =   RED( cs )*SC;
 78:   gs = GREEN( cs )*SC;
 79:   bs =  BLUE( cs )*SC;
 80:
 81:   re =   RED( ce )*SC;
 82:   ge = GREEN( ce )*SC;
 83:   be =  BLUE( ce )*SC;
 84:
 85:   for ( x=0; x<=dx; x++ ) bg[0][x] = br[0][x] = bb[0][x] = 0;
 86:
 87:   w = gx*xs+gy*ys;
 88:   d0 = gx*xe+gy*ye-w;/* 絶対値をとる必要はない(領域内の点に対するd と常に同符号) *
 89:   for ( x=0; x<=dx; x++ ) xx[x] = gx*x;
 90:   lc = 0;
 91:   lb = 1;
 92:   for ( y=0; y<=dy; y++ ) {
 93:     lc = 1-lc;
 94:     lb = 1-lb;
 95:
 96:     cg = cr = cb = 0;
 97:     for ( x=0; x<=dx; x++ ) bg[lb][x] = br[lb][x] = bb[lb][x] = 0;
 98:
 99:     yy = gy*y-w;
100:     for ( x=0; x<=dx; x++ ) {
101:       d = xx[x]+yy;
102:       cr += ( ((d0-d)*rs+d*re)/d0 + br[lc][x] );
103:       cg += ( ((d0-d)*gs+d*ge)/d0 + bg[lc][x] );
104:       cb += ( ((d0-d)*bs+d*be)/d0 + bb[lc][x] );
105:
106:       r = cr/SC; cr -= r*SC;
107:       g = cg/SC; cg -= g*SC;
108:       b = cb/SC; cb -= b*SC;
109:
110:       slbuf[x] = RGB( r, g, b );
111:
112:       br[lb][x] += cr/8;
```

▶僕の見る限りではビデオボード(CZ-6BV1)のピンは6本しかないように見えるが，気のせいだろうか。
今田智宣(16)兵庫県

```
113:      bg[lb][x] += cg/8;
114:      bb[lb][x] += cb/8;
115:      br[lb][ (x>0)?(x-1):(x) ] += cr/4;
116:      bg[lb][ (x>0)?(x-1):(x) ] += cg/4;
117:      bb[lb][ (x>0)?(x-1):(x) ] += cb/4;
118:      br[lb][ (x<dx)?(x+1):(x) ] += cr/8;
119:      bg[lb][ (x<dx)?(x+1):(x) ] += cg/8;
120:      bb[lb][ (x<dx)?(x+1):(x) ] += cb/8;
121:      cr /= 2;
122:      cg /= 2;
123:      cb /= 2;
124:    }
125:    put( x1, y1+y, x1+dx, y1+y, slbuf, sizeof(short)*(dx+1) );
126:  }
127:  return;
128: }
```

リスト7

```
================  crush.c  ==================
  1: /****** ランダム・フラクタルによる自然物（？）の表現 ******/
  2:
  3: #include  <stdlib.h>
  4: #include  <basic0.h>
  5: #include  <graph.h>
  6: #include  <math.h>
  7: #include  "map.h"
  8:
  9: #define GAUSS(X)  ((X)*(rand()-rand()+rand()-rand()+rand()-rand()+rand
 and())/(4*65536))
 10:          /* 疑似正規乱数 */
 11:
 12: #define M 128      /* ランダム・フラクタル配列のサイズ */
 13: #define R 7        /* 再帰レベル：log M */
 14:
 15: int D[M+1][M+1];  /* ランダム・フラクタル配列 */
 16: int L[R+1];       /* 減衰係数 */
 17: int S[R+1];       /* 分割面の大きさ */
 18:
 19: void  rf( n, x, y )
 20: unsigned int  n, x, y;
 21: {
 22:   unsigned int  s, s2;
 23:
 24:   s = S[n];
 25:   s2 = s/2;
 26:
 27:   while ( D[y   ][x+s2]==0 )
 28:     D[y   ][x+s2] = (D[y   ][x   ]+D[y   ][x+s])/2 + GAUSS(L[n]);
 29:   while ( D[y+s ][x+s2]==0 )
 30:     D[y+s ][x+s2] = (D[y+s][x   ]+D[y+s][x+s])/2 + GAUSS(L[n]);
 31:   while ( D[y+s2][x   ]==0 )
 32:     D[y+s2][x   ] = (D[y   ][x   ]+D[y+s][x   ])/2 + GAUSS(L[n]);
 33:   while ( D[y+s2][x+s ]==0 )
 34:     D[y+s2][x+s ] = (D[y   ][x+s]+D[y+s][x+s])/2 + GAUSS(L[n]);
 35:
 36:   while ( D[y+s2][x+s2]==0 )
 37:     D[y+s2][x+s2] = (D[y][x+s2]+D[y+s][x+s2]+D[y+s2][x]+D[y+s2][x+s])/4
 + GAUSS(L[n]);
 38:
 39:   if ( s>2 ) {
 40:     rf( n+1, x    , y     );
 41:     rf( n+1, x+s2, y     );
 42:     rf( n+1, x    , y+s2 );
 43:     rf( n+1, x+s2, y+s2 );
 44:   }
 45:   return;
 46: }
 47:
 48: FUNC  random_fractal( dummy )
 49: DUMMY dummy;
 50: /* float l0, b; (int seed(, d00, d01, d10, d11 )) */
 51: {
 52:   double  l0, b;
 53:   int     seed=1234, d00=0, d01=0, d10=0, d11=0;
 54:   int     i, x, y;
 55:
 56:   ARGSET( dummy );
 57:   l0=FVALUE(1);
 58:   b=FVALUE(2);
 59:   if ( ARGC>2 ) seed=IVALUE(3); /* 以下は省略可 */
 60:   if ( ARGC==7 ) {
 61:     d00=IVALUE(4);
 62:     d01=IVALUE(5);
 63:     d10=IVALUE(6);
 64:     d11=IVALUE(7);
 65:   }
 66:   srand( seed );
 67:   for ( i=0; i<=R; i++ ) {  /* 減衰係数を設定する */
 68:     L[i] = (int)(l0*exp(-b*(double)i));
 69:     S[i] = (M>>i);           /* 再帰分割面の大きさ */
 70:   }
 71:   for ( y=0; y<=M; y++ ) {  /* 初期化 */
 72:     for ( x=0; x<=M; x++ ) {
 73:       D[y][x]=0;
 74:     }
 75:   }
 76:   D[0][0]=d00;  /* 初期値 */
 77:   D[0][M]=d01;
 78:   D[M][0]=d10;
 79:   D[M][M]=d11;
 80:
 81:   rf( 0,0,0 );  /* 再帰分割によるランダム・フラクタル */
 82:
 83:   return ( 0 );
 84: }
 85:
 86: extern unsigned char  OUTOF_SCREEN[];
 87:
 88: FUNC  crush( dummy )
 89: DUMMY dummy;
 90: /* int x1, y1, x2, y2; float k */
 91: {
 92:   int     x1, y1, x2, y2;
 93:   int     idx, idy;
 94:   double  k;
 95:   int     x, y, yf, wy, tmp;
 96:   double  d, d00, d01, d11, d10;
 97:   double  dx, dy, ix, iy;
 98:   int     ix0, ix1, iy0, iy1;
 99:   static unsigned short slbuf0[ N_PIXEL ], slbuf1[ N_PIXEL ];
100:
101:   ARGSET( dummy );
102:   x1=IVALUE(1);
103:   y1=IVALUE(2);
104:   x2=IVALUE(3);
105:   y2=IVALUE(4);
106:   k=FVALUE(5);
107:
108:   if ( x2<x1 ) {
109:     tmp=x1;
110:     x1=x2;
111:     x2=tmp;
112:   }
113:   if ( y2<y1 ) {
114:     tmp=y1;
115:     y1=y2;
116:     y2=tmp;
117:   }
118:   idx = x2-x1;
119:   idy = y2-y1;
120:   dx = (double)idx;
121:   dy = (double)idy;
122:   wy = idy+1;
123:
124:   if ( x1<0 || x2>=N_PIXEL || y1<0 || y2>=N_PIXEL ) {
125: #ifdef  __GNUC__
126:     asm( "  lea.l _OUTOF_SCREEN,a1" );
127: #else
128: #asm
129:   lea.l _OUTOF_SCREEN,a1
130: #endasm
131: #endif
132:     return ( 1 );
133:   }
134:
135:   for ( x=0; x<=idx; x++ ) {
136:     get( x1+x, y1, x1+x, y2, slbuf0, sizeof(short)*wy );
137:     ix = (double)x*(double)M/dx;  /* 補間に使う座標 */
138:     ix0 = (int)ix;                /* 整数部（切り捨て）*/
139:     ix1 = ix0+1;                  /* 整数部（切り上げ） */
140:     ix -= (double)ix0;            /* 小数部 */
141:     for ( y=0; y<=idy; y++ ) {
142:       iy = (double)y*(double)M/dy;/* 補間に使う座標 */
143:       iy0 = (int)iy;              /* 整数部（切り捨て）*/
144:       iy1 = iy0+1;                /* 整数部（切り上げ） */
145:       iy -= (double)iy0;          /* 小数部 */
146:
147:       d00 = (double)D[iy0][ix0];  /* ランダムフラクタル格子を */
148:       d01 = (double)D[iy0][ix1];  /* 線形補間する */
149:       d10 = (double)D[iy1][ix0];
150:       d11 = (double)D[iy1][ix1];
151:       d = (1.0-iy)*((1.0-ix)*d00+ix*d01)+iy*((1.0-ix)*d10+ix*d11);
152:
153:       yf = y+(int)(k*d);
154:       if ( yf>idy ) yf=idy;
155:       if ( yf<0 ) yf=0;
156:       slbuf1[y] = slbuf0[yf];
157:     }
158:     put( x1+x, y1, x1+x, y2, slbuf1, sizeof(short)*wy );
159:   }
160:   return ( 0 );
161: }
```

リスト8

```
================  main.c  ==================
  1: /****** パラメータ受け渡し用仮変数の実体 ******/
  2:
  3: unsigned short  *par;         /* 一時的な引数リスト */
  4: unsigned short  *ary[10+1]; /* 一時的な配列リスト：X-BASIC の引数は最大 10
 個 */
  5:
  6: /****** コンパイラを通すためのダミー （ 実行されない ） ******/
  7:
  8: void  main()
  9: {
 10: }
```

X68000CARD.FNC用カードゲーム

赤黒(SPEED) BLACK JACK

Sugou Masaru
菅生 勝
Mounai Toshiyuki
毛内 俊行

CARD.FNCを使ったゲームシリーズ第3，4弾。今回はかなりポピュラーなゲーム，赤黒（SPEED）とBLACK JACKの登場だ。せっかくのCARD.FNC，目一杯使ってみようじゃないか。まだまだネタはつきないぞ。みんな，どんどん投稿してくれ。

リアルタイムカードゲーム

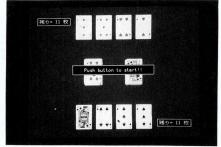
CARD.FNCを使ってトランプゲームを作ってみました。6月号の付録ディスクのおかげで全国のユーザーがCARD.FNCを持っていることと思います。CARD.FNCはOh!X 5月号で発表されたX-BASIC用のカードゲーム支援関数群です。カードのグラフィックデータを内蔵しているので，誰にでも手軽にカードゲームを作ることができます。

今回発表するゲームは「赤黒」です。知っていますか？　もしくは「SPEED」と呼ばれることのほうが多いかもしれません。トランプを酷使するゲームとして有名ですね。私は小学校の頃よくやっていたもんです。

ルール

一応，ルールを解説しておきます。カードを赤と黒の2つに分けておき，プレイヤーは黒のカードを持っています。画面には手札のうちの4枚ずつが表示されていますね。マウスのボタンを押すとゲームスタートです。

ゲームが始まると手札のうちの1枚ずつが画面中央に現れます。これらの上に4枚の台札のうち数上がりまたは数下がりのシーケンスを組むものを重ねていきます。台札の空いたところには手札から新しいカードが補充され，先に手持ちのカードすべてをなくしたほうの勝ちになります。なお，台札はマウスボタンの左右でどちらの山に積んでいくかを指定します。台札を出す際にはスート（記号）や左右の位置は関係ありません。

双方とも出せるカードがなくなったら手札から新しいカードを台札として1枚ずつ山に積みます。カードが出せる状態なら，いつでもカードを出すことができます。要するに手の速い人が勝つという，トランプには珍しいリアルタイムゲームなのです。

双方の手札がなくなった場合は，残った台札の数で勝敗が決まります。同数のときは数字の合計が少ないほうが勝ちとなります。そして，コンピュータかプレイヤーかどちらかが3勝した時点で1ゲーム終了です。

赤黒

＊　＊　＊

できるだけ短いプログラムにしようと思ったのですが，プログラムを作るのが下手なもので，やはり300行を少し超えてしまいました。

プログラム中のシャッフルルーチンは5月号の「99」のものをそのまま使わせてもらいました。　(M.S.)

BLACK JACK

こんにちは。CARD.FNCはもうBASICに組み込みましたか？　先月のHEARTに引き続き，今月もSPEEDが投稿され，予想以上の反響に驚いています。なにを隠そう実は私もSPEEDを作って編集室へ持っていったのですが「投稿が来てるよ」の一言でボツになってしまいました。「ガーン！　しまったぁ……」といってもあとの祭りアフターフェスティバル。でも，そこで引き下がっては男の名折れ。このショックをエネルギーに変えて，わずか3日で作ってしまったのがこのブラックジャックです。

BLACK JACK

ルールの説明

人数は2人以上，使うカードはジョーカーを除く52枚です。札には，次に示すようにそれぞれ点数がついています。

A……1点または11点（任意）
絵札……10点
数札……数字どおりの点数

ゲームは最初に，ベット（賭け金）を場に出して始まります。ディーラーがベットを出した人のところにカードを配ります。カードは1枚目を伏せて，2枚目は表にして配られます。カードが2枚配られたら，伏せてある1枚目のカードを自分だけで見て，1枚目と2枚目のカードの点数を足し合わせます。この点数がハンドの基本点となります。このあとディーラーがもう1枚カードを引くか聞いてくるので，必要なら1枚もらいます。もちろん，その3枚目の札もハンドの点数に加算されます。

ここで注意しなければいけないのは，ハンドの点数は21点を越えてはいけないということです。ハンドの点数が21点を1点でもオーバーしたら，ハンドの点数は0点になってしまいます。たとえばハンドの点数が19点もあるのに，もう1枚カードを引くということは，危険きわまりないことだということを念頭において，3枚目を取るかどうか決めます。

最後にディーラーが3枚目を取るかどう

か決めてから，お互いに伏せてあった１枚目のカードを見せ合い，強いハンドを持っていたほうが場のベットをもらいます。ディーラーはゲームのあいだじゅう変わることはありません。

なお，本来ブラックジャックでは21点を作ったときのハンドのかたちによって，賭け金の倍払いや３倍払いというルールがあるはずなのですが，誰に聞いてもいうことが全然違うので，今回は倍払いルールはつけませんでした。

プログラムでは，コンピュータがディーラーになってあなたと１対１で対戦します。最初にお互いにコインが５枚配られ，１ゲームにつき１枚ずつベットとして支払います。ゲームは相手のコインをすべて巻き上げたほうの勝ちになります。ゲームはほとんど自動的に進行します。途中，３枚目のカードを引くかどうか聞いてくるので，マウスを操作して，YESまたはNOのところでクリックしてください。

最初はコイン20枚くらいで対戦させようと思っていたのですが，あまりにもコインが多いとゲームが終了するのに１～２時間もかかってしまって話になりませんでした。コインの枚数を変えたい人はリストの行番号90の変数kazuの値を変えてみてください。

＊　＊　＊

99から始まって，HEART，SPEEDと，すでにCARD.FNC対応のゲームは４本も揃ってしまいました。しかし，トランプゲームは奥が深い。まだまだたくさんのゲームがX68000で動く日を待っています。読者の皆さんのなかで「我こそは」と思う人は，どんどんゲームを作って投稿しましょう。待ってます。　(T.M)

リスト１　赤黒

```
  10 /*カードゲーム赤黒(要 CARD.FNC)
  20 /*
  30 dim int aka(25),kuro(25)
  40 dim int aka_te(3),kuro_te(3)
  50 int l_card,r_card,r_od=0,b_od=0
  60 int r_sheet=26,b_sheet=26,vic=0,def=0
  70 int mx,my,x,y,rb,lb
  80 int man=385,com=32,see=0,pass=0
  90 int g_end=0,begin=1,endf=0
 100 str level="t20018rrrrrrrrrr"
 110 init():mml_init():card_init()
 120 /*
 130 while (endf=0)/*メインプログラム
 140   nv_init():shuffle():card_place()
 150   while (g_end=0)
 160     message(1,0):open_card()
 170     while(begin=1)
 180       man_play():com_play()
 190     endwhile
 200   endwhile
 210 endwhile
 220 end
 230 /*
 240 func man_play()/*人間
 250   int ps,ch
 260   mspos(mx,my):msstat(x,y,lb,rb)
 270   if (lb or rb) then {
 280     ps=(mx-138) ¥ 60:pass=0
 290     if (lb=-1) then {
 300       kuro_nokori(cd_check(ps,0,kuro_te(ps),man),ps)
 310     } else {
 320       kuro_nokori(cd_check(ps,1,kuro_te(ps),man),ps)
 330     }
 340   }
 350 endfunc
 360 /*
 370 func com_play()/*コンピュータ
 380   int i,cd,ch
 390   if (m_stat(1)=0 and pass=0) then {
 400     for i=0 to 3
 410       if (aka_nokori(cd_check(see,0,aka_te(see),com),see))
then return()
 420       if (aka_nokori(cd_check(see,1,aka_te(see),com),see))
then return()
 430       see=(see+1) mod 4
 440     next:pass=1
 450     if (vic_check()) then return()
 460     if (man_check()=0) then {
 470       if (r_od=26 and b_od=26) then judge()
 480     }
 490   }
 500 endfunc
 510 /*
 520 func int aka_nokori(ch;int,ps;int)/*カードが残っているか
 530   if (ch=1) then {
 540     sheet_dsp(0):m_play(1)
 550     if (r_od<26) then card_put(ps,0) else aka_te(ps)=-1
 560     if (r_sheet=4) then reverse()
 570     return(1)
 580   }
 590   return(0)
 600 endfunc
 610 /*
 620 func kuro_nokori(ch;int,ps;int)/*カードが残っているか
 630   int dm
 640   if (ch=1) then {
 650     sheet_dsp(1)
 660     if (b_od<26) then card_put(ps,1) else {
 670       kuro_te(ps)=-1:dm=vic_check()
 680     }
 690   }
 700 endfunc
 710 /*
 720 func sheet_dsp(sw;int)/*残りの枚数を表示
 730   switch sw
 740     case 0 :r_sheet=r_sheet-1
 750             locate 10,3:print using "##";r_sheet
 760             break
 770     case 1 :b_sheet=b_sheet-1
 780             locate 55,28:print using "##";b_sheet
 790   endswitch
 800 endfunc
 810 /*
 820 func int man_check()/*出せるカードがあるか
 830   int i,cd
 840   for i=0 to 3
 850     cd=(kuro_te(i)-1) mod 13+1
 860     if (l_card=cd+1 or l_card=cd-1) then return(1)
 870     if (r_card=cd+1 or r_card=cd-1) then return(1)
 880     if (ace_king(l_card,cd) or ace_king(r_card,cd)) then re
turn(1)
 890   next
 900   begin=0:return(0)
 910 endfunc
 920 /*
 930 func int cd_check(n;int,lr;int,c_num;int,pc;int)/*出せるカ
ードか
 940   int cd,c1,c2
 950   cd=(c_num-1) mod 13+1
 960   switch lr
 970     case 0:if (l_card=cd+1 or l_card=cd-1) then {
 980                card_clr(n,pc,0,c_num):l_card=cd:return(1)}
 990              if (ace_king(l_card,cd)) then {
1000                card_clr(n,pc,0,c_num):l_card=cd:return(1)}
1010              break
1020     case 1:if (r_card=cd+1 or r_card=cd-1) then {
1030                card_clr(n,pc,1,c_num):r_card=cd:return(1)}
1040              if (ace_king(r_card,cd)) then {
1050                card_clr(n,pc,1,c_num):r_card=cd:return(1)}
1060   endswitch
1070   return(0)
1080 endfunc
1090 /*
1100 func int ace_king(lr_c;int,cd;int)/*場がAかKの場合
1110   if (lr_c=1 or lr_c=13) then {
1120     if (lr_c=1 and cd=13) or (lr_c=13 and cd=1) then return
(1)
1130   }
1140   return(0)
1150 endfunc
1160 /*
1170 func card_clr(n;int,pc;int,lr;int,c_num;int)/*カードを出す
1180   int w
1190   w=n*60+143
1200   window(w,pc,w+48,pc+96):wipe()
1210   window(0, 0, 511,  511)
1220   c_put(lr*121+172+(rnd()*10-5),208+(rnd()*10-5),c_num)
1230 endfunc
1240 /*
1250 func card_put(n;int,sw;int)/*手のカードを追加
1260   int ps
1270   ps=n*60+143
1280   switch sw
1290     case 0:c_put(ps,com,aka(r_od))
1300              aka_te(n)=aka(r_od):r_od=r_od+1
1310              break
1320     case 1:c_put(ps,man,kuro(b_od))
1330              kuro_te(n)=kuro(b_od):b_od=b_od+1
1340   endswitch
1350 endfunc
1360 /*
1370 func int vic_check()/*カードがなくなれば勝ち
1380   if (r_sheet=0) then defeat():loop_out():return(1) else {
1390     if (b_sheet=0) then victory():loop_out():return(1)
1400   }
1410   return(0)
1420 endfunc
1430 /*
1440 func judge()/*カードが残っている場合の判定
1450   int i,j,rs,bs,rc,bc
1460   for i=0 to 3
1470     if (aka_te(i)<>-1) then {
1480       rs=rs+1:rc=rc+(( aka_te(i)-1) mod 13+1)
1490       j=i*60+143:c_put(j,com,aka_te(i))
1500     }
1510     if (kuro_te(i)<>-1) then {
1520       bs=bs+1:bc=bc+((kuro_te(i)-1) mod 13+1)
1530     }
1540   next
1550   if (rs<bs) then defeat() else {/*枚数での判定
1560     if (rs>bs) then victory() else {
1570       if (rs=bs) then {
1580         if (rc<bc) then defeat()/*数字での判定
1590         if (rc>bc) then victory()
1600         if (rc=bc) then message(4,0)
1610   }}}
1620   loop_out()
1630 endfunc
```

```
1640 /*
1650 func victory()/*勝ち
1660   vic=vic+1
1670   level=left$(level,len(level)-vic):mml_init()
1680   if (vic=3) then {
1690     game_end(vic,def,0):message(5,0):question()
1700   } else message(3,0)
1710 endfunc
1720 /*
1730 func defeat()/*負け
1740   def=def+1
1750   if (def=3) then {
1760     game_end(vic,def,1):message(8,0):question()
1770   } else message(2,0)
1780 endfunc
1790 /*
1800 func question()/*質問
1810   message(7,1):msarea(292,236,373,259)
1820   repeat
1830     mspos(mx,my):msstat(x,y,lb,rb)
1840   until(lb=-1)
1850   if (mx<333) then {
1860     loop_out():level="t20018rrrrrrrrrr"
1870     mml_init():msarea(137,379,377,487)
1880     vic=0:def=0:mess_clr()
1890   } else loop_out():endf=1
1900 endfunc
1910 /*
1920 func loop_out()/*ループを抜ける場合(ゲーム再会)
1930   pass=1:begin=0:g_end=1:wipe()
1940 endfunc
1950 /*
1960 func open_card()/*赤黒1枚ずつ場に表示
1970   if (r_od<26) then {
1980     c_put(172,208,aka(r_od))
1990     l_card=(aka(r_od)-1) mod 13+1
2000     r_od=r_od+1:sheet_dsp(0)
2010     if (r_sheet=4) then reverse()
2020   }
2030   if (b_od<26) then {
2040     c_put(293,208,kuro(b_od))
2050     r_card=(kuro(b_od)-1) mod 13+1
2060     b_od=b_od+1:sheet_dsp(1)
2070   }
2080   m_play(1):begin=1:pass=0
2090 endfunc
2100 /*
2110 func shuffle()/*シャッフル
2120   int i,j,k,l
2130   message(6,1)
2140   for i=0 to 50
2150     j=rnd()*25:k=rnd()*25
2160     l=aka(j):aka(j)=aka(k):aka(k)=l
2170     j=rnd()*25:k=rnd()*25
2180     l=kuro(j):kuro(j)=kuro(k):kuro(k)=l
2190   next
2200   mess_clr()
2210 endfunc
2220 /*
2230 func init()/*初期設定
2240   int i,j
2250   screen 1,1,1,1:console ,,0:vpage(13)
2260   mouse(4):mouse(1):palet(1,0):apage(3)
2270   fill(0,0,511,511,8)
2280   fill( 24, 40,127, 72,1)
2290   fill(384,440,487,472,1)
2300   box( 26, 42,125, 70,15)
2310   box(386,442,485,470,15)
2320   locate  4, 3:print "残り= 26 枚"
2330   locate 49,28:print "残り= 26 枚"
2340   apage(1):fill(130,227,380,267,1)
2350   box(132,229,378,265,15):apage(2)
2360   msarea(137,379,377,487)
2370   for i=0 to val(right$(time$,2)):rnd():next
2380 endfunc
2390 /*
2400 func mml_init()/*FMの初期化
2410   m_init():m_alloc(1,50)
2420   m_assign(1,1):m_trk(1,level)
2430 endfunc
2440 /*
2450 func nv_init()/*変数の初期化
2460   r_od=0:b_od=0:g_end=0:begin=1
2470   see=0:pass=0:r_sheet=26:b_sheet=26
2480 endfunc
2490 /*
2500 func card_init()/*赤と黒に分ける
2510   int i,j,sr=1,sb=0
2520   repeat
2530     for i=1 to 13
2540       aka(j) =sr*13+i
2550       kuro(j)=sb*13+i:j=j+1
2560     next
2570     sr=sr+1:sb=sb+3
2580   until(j=26)
2590 endfunc
2600 /*
2610 func card_place()/*最初の4枚を表示
2620   int i,j=3
2630   for i=0 to 3
2640     card_put(i,1):card_put(j,0):j=j-1
2650   next
2660 endfunc
2670 /*
2680 func reverse()/*カードを裏向ける
2690   int i,j
2700   for i=0 to 3
2710     j=i*60+143:c_put(j,com,0)
2720   next
2730 endfunc
2740 /*
2750 func message(n;int,rt;int)/*メッセージ表示
2760   str m
2770   switch n
2780     case 1:m="   Push button to start!!   ":break
2790     case 2:m="  コンピューターの勝ちです  ":break
2800     case 3:m="       貴方の勝ちです       ":break
2810     case 4:m="          引き分け          ":break
2820     case 5:m="    貴方は赤黒の天才です    ":break
2830     case 6:m="    シャッフルしています    ":break
2840     case 7:m="もう一度やりますか[Y][N]":break
2850     case 8:m="         残念でした         "
2860   endswitch
2870   vpage(14):locate 18,15:print m
2880   if rt then return()
2890   button_on():mess_clr():button_off()
2900 endfunc
2910 /*
2920 func game_end(v;int,d;int,pc;int)/*結果表示
2930   str p,c,m,vd(1)={"勝ち","負け"}
2940   p=chr$(130)+chr$(79+v)
2950   c=chr$(130)+chr$(79+d)
2960   m=p+"勝"+c+"敗で貴方の"+vd(pc)+"です"
2970   vpage(14):locate 20,15:print m
2980   button_on():mess_clr():button_off()
2990 endfunc
3000 /*
3010 func button_on()
3020   repeat:msstat(x,y,lb,rb):rnd():until(lb=-1)
3030 endfunc
3040 /*
3050 func mess_clr()/*メッセージ消去
3060   locate 18,15:print space$(28):vpage(13)
3070 endfunc
3080 /*
3090 func button_off()/*ボタンが離されるまで
3100   repeat:msstat(x,y,lb,rb):until(lb=0)
3110 endfunc
```

リスト2 BLACK JACK

```
10 /*========================================
20 /*     究極の card.fnc 用カードゲーム
30 /*
40 /*     ブラックジャックだよーん!!
50 /*
60 /*            by T.Mounai
70 /*========================================
80 /*
90 int pnum,ct,kazu=5,i,round
100 str pasa="@59v15c",win="@56v15o4l32cegb8g"
110 str lost="@53v15o4c<c",pico="@56v15o4l20aeae"
120 str wake="@56v15c",tin="@50v15o6l64cc"
130 str nam
140 dim int card(51),hand(5),kane(1)
150 dim int syohai(1)={0,0}
160 dim str plr(2)={"うさぎさん","きつねさん","ねこさん"}
170 dim char coin(127)={
180   +0,  0, 13,221,221,  0,  0,  0,
190   +0,  0,220,204,204,208,  0,  0,
200   +0, 13,205,221,221,205,  0,  0,
210   +0, 13,205,220,221,205,  0,  0,
220   +0,220,221,204,204,220,208,  0,
230   +0,220,220,220,221,220,208,  0,
240   +0,220,220,220,221,220,208,  0,
250   +0,220,221,204,205,220,208,  0,
260   +0,220,221,220,220,220,208,  0,
270   +0,220,221,220,220,220,208,  0,
280   +0,220,220,204,205,220,208,  0,
290   +0, 13,205,220,221,205,  0,  0,
300   +0, 13,205,221,221,205,  0,  0,
310   +0,  0,220,204,204,208,  0,  0,
320   +0,  0, 13,221,221,  0,  0,  0,
330   +0,  0,  0,  0,  0,  0,  0,  0
340 }
350 dinit()
360 while 1
370   aite():for i=0 to 1:kane(i)=0:syohai(i)=0:next
380   kinit():round=0
390   repeat
400     cls:wipe()
410     sfl():betset():round=round+1
420     deal():cdmore()
430     hantei(syobu())
440   until kane(0)=0 or kane(1)=0
450   gover()
460 endwhile
470 end
480 /*
490 func aite()               /* 対戦相手の決定
500 apage(3):fill(8,112,87,127,2)
510 pnum=rand() mod 3:nam=plr(pnum)
520 symbol(8,112,nam,1,1,1,15,0):apage(1)
530 endfunc
540 /*
550 func kinit()              /* コインの表示
560 int i,j
570 for i=0 to 1
580   for j=0 to kazu-1
590     kput(i,j,1)
600   next
610 next
620 endfunc
630 /*
640 func oto(a;str)            /* 効果音
650 m_init():m_trk(1,a):m_play()
660 endfunc
```

```
670 /*
680 func wtm(n)                    /* 時間待ち
690 int i
700 for i=0 to n*500:next
710 endfunc
720 /*
730 func dinit()                   /* 画面初期化
740 randomize(val(mid$(time$,4,2))*100+val(right$(time$,2)))
750 m_assign(1,1):m_alloc(1,100)
760 screen 1,1,1,1:console ,,0:palet(1,0)
770 apage(3)
780 fill(0,0,511,511,2)
790 fill(120,100,391,411,8)
800 symbol(31,31,"ＢＬＡＣＫ　ＪＡＣＫ",1,2,2,0,0)
810 symbol(30,30,"ＢＬＡＣＫ　ＪＡＣＫ",1,2,2,15,0)
820 symbol(8,304,"あなた",1,1,1,15,0)
830 mouse(1):mouse(4):setmspos(343,390)
840 endfunc
850 /*
860 func cdput(x,y,n)             /* 札の画面表示1
870 c_put(x*60+170,y*180+120,n)
880 oto(pasa)
890 endfunc
900 /*
910 func oc_card(pl,n,of)         /* 札の画面表示2
920 int cn
930 if of=1 then cn=hand(pl*3+n) else cn=0
940 cdput(n,pl,cn)
950 endfunc
960 /*
970 func sfl()                     /* シャッフル
980 int a,b,c,i
990 for i=0 to 5:hand(i)=0:next
1000 for i=0 to 51:card(i)=i+1:next
1010 for i=0 to 99
1020   a=rnd()*52:b=rnd()*52
1030   c=card(a):card(a)=card(b):card(b)=c
1040 next
1050 endfunc
1060 /*
1070 func deal()                   /* 最初にカードを配る
1080 int i=0,j=0
1090 repeat
1100   if i=2 then i=i+1
1110   hand(i)=card(j)
1120   i=i+1:j=j+1
1130 until i=5
1140 oc_card(0,0,0):wtm(1)
1150 oc_card(1,0,0):wtm(1)
1160 oc_card(0,1,1):wtm(1)
1170 oc_card(1,1,1):wtm(10)
1180 oc_card(1,0,1):ct=4
1190 endfunc
1200 /*
1210 func askpt(n)                 /* 点数評価
1220 int a
1230 if n=0 then return(-1)
1240 a=n mod 13
1250 switch a
1260   case  0:a=10:break
1270   case 11:a=10:break
1280   case 12:a=10:break
1290   default:break
1300 endswitch
1310 return(a)
1320 endfunc
1330 /*
1340 func handpt1(pl)              /* ハンドの評価(a=1)
1350 int a,i,n=0
1360 for i=0 to 2
1370   a=askpt(hand(pl*3+i))
1380   if a=-1 then break
1390   n=n+a
1400 next
1410 if n>21 then n=0
1420 return(n)
1430 endfunc
1440 /*
1450 func handpt2(pl)              /* ハンドの評価(a=10)
1460 int a,i,n=0,af=0
1470 for i=0 to 2
1480   a=askpt(hand(pl*3+i))
1490   if a=-1 then break
1500   if a=1 and af=0  then a=11:af=1
1510   n=n+a
1520 next
1530 if n>21 then n=0
1540 return(n)
1550 endfunc
1560 /*
1570 func cdmore()                 /* カードの追加
1580 if yesno() then hand(5)=card(ct):ct=ct+1:oc_card(1,2,1)
1590 if dcom() then hand(2)=card(ct):oc_card(0,2,1)
1600 endfunc
1610 /*
1620 func yesno()                  /* もう1枚引く(人間)
1630 int a,x,y,l,r
1640 msg("もう1枚引く？")
1650 apage(0)
1660 fill(310,383,338,401,3)
1670 fill(350,383,378,401,5)
1680 locate 39,24:print"yes  no"
1690 repeat
1700   mspos(x,y):l=rnd()
1710   a=point(x,y)
1720   if a=4 or a=15 then a=0
1730   msstat(x,y,l,r)
1740 until l+r<>0 and a<>0
1750 apage(1)
1760 switch a
1770   case 3:a=1:break
1780   case 5:a=0:break
1790 endswitch
1800 cls:msg("")
1810 fill(182,236,296,276,0)
1820 return(a)
1830 endfunc
1840 /*
1850 func dcom()                   /* もう1枚引く(コンピュータ)
1860 int a,p1,p2,pt,ps,rr
1870 p1=handpt1(0):p2=handpt2(0)
1880 if p1>p2 then pt=p1 else pt=p2
1890 rr=rand() mod 5+3:ps=21-pt
1900 if ps>7 then return(1)
1910 if ps<3 then return(0)
1920 if ps>rr then return(1)
1930 return(0)
1940 endfunc
1950 /*
1960 func syobu()                  /* 勝負！！
1970 int p1,p2,m,c
1980 msg("   勝  負！")
1990 oto(pico):wtm(20):msg("")
2000 oc_card(0,0,1):p1=handpt1(0):p2=handpt2(0)
2010 if p1<p2 then c=p2 else c=p1
2020 p1=handpt1(1):p2=handpt2(1)
2030 if p1<p2 then m=p2 else m=p1
2040 locate 45,10:print c:wtm(10)
2050 locate 45,20:print m:wtm(5)
2060 if c>m then return(0)
2070 if c<m then return(1)
2080 return(-1)
2090 endfunc
2100 /*
2110 func msg(m;str)               /* メッセージ表示
2120 if m="" then {
2130  apage(0)
2140  wipe()
2150  apage(1)
2160 } else {
2170   apage(0)
2180   fill(190,244,316,276,4)
2190   symbol(198,252,m,1,1,1,15,0)
2200   apage(1)
2210 }
2220 endfunc
2230 /*
2240 func hantei(n)                /* 勝利判定
2250 switch n
2260   case  0:msg("   私の勝ち")  :oto(lost):wtm(10)
2270           kput(0,kane(0),1):betclr(0):wtm(3)
2280           kput(0,kane(0),1):betclr(1)
2290           syohai(0)=syohai(0)+1:break
2300   case  1:msg(" あなたの勝ち"):oto(win) :wtm(10)
2310           kput(1,kane(1),1):betclr(0):wtm(3)
2320           kput(1,kane(1),1):betclr(1)
2330           syohai(1)=syohai(1)+1:break
2340   case -1:msg("   引き分け")  :oto(wake):wtm(10)
2350           kput(0,kane(0),1):betclr(0):wtm(3)
2360           kput(1,kane(1),1):betclr(1):break
2370 endswitch
2380 wtm(20):msg("")
2390 endfunc
2400 /*
2410 func kput(pl,n,f)             /* コイン表示
2420 int x,y
2430 apage(2)
2440 switch pl
2450   case 0:y=128:break
2460   case 1:y=320:break
2470 endswitch
2480 x=(n mod 5)*16+8
2490 y=y+(n/5)*16
2500 if f then {
2510   put(x,y,x+15,y+15,coin)
2520   oto(tin)
2530   kane(pl)=kane(pl)+1
2540 } else {
2550   fill(x,y,x+15,y+15,0)
2560   kane(pl)=kane(pl)-1
2570 }
2580 apage(1)
2590 endfunc
2600 /*
2610 func betset()                 /* 賭け金を表示
2620 kput(0,kane(0)-1,0):apage(2):oto(tin)
2630 put(96,112,111,127,coin):apage(1):wtm(3)
2640 wtm(3):kput(1,kane(1)-1,0):apage(2):oto(tin)
2650 put(96,304,111,319,coin):apage(1):wtm(3)
2660 endfunc
2670 /*
2680 func betclr(n)                /* 賭け金を消す
2690 int y
2700 if n=0 then y=112 else y=304
2710 apage(2):fill(96,y,111,y+15,0):apage(1)
2720 endfunc
2730 /*
2740 func gover()                  /* ゲームーオーバー
2750 int a,win,lost,ave,l,r,x,y
2760 str winner
2770 cls:apage(0):fill(150,150,361,361,3)
2780 win=syohai(1):lost=syohai(0):winner="あなた"
2790 if kane(1)=0 then {
2800   a=win:win=lost:lost=a
2810   winner=plr(pnum)
2820 }
2830 ave=(win*100)/(round-(round-win-lost))
2840 locate 22,11:print winner;"が勝ちました"
2850 locate 22,13:print"ラウンド数";round
2860 locate 21,15:print win;"勝";lost;"敗";round-win-lost;"分"
2870 locate 22,17:print"勝率は";ave;"% です"
2880 locate 22,20:print"click mouse button!!"
2890 repeat:msstat(x,y,l,r):until l+r<>0
2900 wipe():cls:apage(2):wipe():apage(1)
2910 endfunc
```

X68000用
風の谷のナウシカ

Ando Masahiro
安藤 正洋

X1/turbo用
ラジオ体操第一

Kamio Souichi
神生 総一

毎日暑い日が続きますね。さて今月は，X68000用には「風の谷のナウシカ」を，そしてX1用にはこの時期ならではの「ラジオ体操第一」です。さすがにもう近所の小学校にはいきづらいでしょうし，朝っぱらからラジオ体操なんて元気はさらさらないでしょうから，これを打ち込んで密かに部屋でラジオ体操してください。

待ちに待ってた安田成美のデビュー曲

かなり前の話（1990年3月号）で恐縮なのですが，「風の谷のナウシカ」が聞きたい，それも安田成美さんが歌ってたやつがいい，てなことをこのページで書いたんです。なんてわがままなお願いなんだろうと思いつつも，催促[3]なんて書いたんですよ。そしたらね，嬉しいじゃありませんか，常連の安藤君が一肌脱いでくれたんですよ。なんといっても6年も前の曲だし，サントラと違ってイメージソングだからテープだって入手困難だったでしょうに……。「参考にしたもの月刊明星1984年7月号付録YOUNG SONG」っていうのが泣かせますよね。

実は4月に投稿していただいてたのですが，諸般の事情により掲載が延ばし延ばしになっていたのです。ごめんなさい。もともとの曲がそんなに派手ではないために，近頃のOPMA SONGSと比べるとちょっと寂しいような気がするかもしれませんが，確かにこういった雰囲気の曲でした。個人的にはバックのオカリナが細野さんらしくてとっても好きです。

今日は泉岳寺からお送りします

「夏」っていったら海や山もいいけど，子供の頃にやったラジオ体操ってのもいいもんですよね。まあ，やってた頃はめんどくさいもんだったように記憶してますけども。スタンプ集めて，最後の日に何かをもらうってのはどこでもやっているものなのでしょうか？　私は花火セットかノートだったように記憶してるんですけど。今現在でラジオ体操をやっている人，何をもらったかはっきり覚えている人は「ラジオ体操・私はこれをもらった」係までよろしくお願いします。

と，ここまで書けば曲は自然に聞こえてきますね？　そう，ラジオ体操です。タイムリーでしょ？　ちなみに第一です。ラジオ体操第二を「ラジオ体操」っていう曲の2番だと思っている人もいるようですが，それは間違いです。作曲者が違うのです。第一は服部正さん，第二は團伊玖磨さんなのです。よく覚えておきましょう。試験に出るかもしれません（んなこたねー）。それでは最近のカラオケブームにあやかって今回は特別に歌詞（？）を載せてしまいましょう。

・腕を前から上に挙げて，のびのびと背伸びの運動から，はい，
・1234567 手足の運動
・12345678
・1234567 腕を回します
・外回し内回し5678
・123456 胸の運動足を開いて
・横振りななめ上5678
・123456 体を横に曲げる運動
・12345678
・123456 はい下に曲げましょう
・3回～手と腰後ろに
・123456 腕を振って体をねじる運動
・左右左右左に大きく78
・右左右左56足を戻して手足の運動
・12345678
・123456 足を横に出してななめ下に
・正面で起こし～78
・～34～腕を振って体を回す運動
・1・34～78
・～34～足を戻して両足跳び
・1・34開いて閉じて開いて閉じて
・～～開いて閉じて手足の運動
・12345678

安田成美

・123456 のびのびと深呼吸
・～34～78～34～78

＊指導／柳川英鷹

注：上記で“～”とあるところでは喋っていません。2拍分ですのでタイミングを取るときに注意してください。同様に“1・34”では“2”の音(1拍分）を出していないことを表しています。慣れないと“123……”を後ろのピアノにあわせて音程を取ってしまいがちですが，練習すればちゃんと歌える（？）ようになります。歯切れよく，元気に，自分を捨て去って歌うのがコツです。ちゃんとテープから採詞したので間違いはないはずです。あとは，「今日は千代田区の宝珍小学校の校庭に2年C組の皆さんに集まってもらいました」のようなノリのものをアタマに付ければ，ほぼ完璧といえます。なお，この曲はX1のMUSIC BASIC用です。音色データはVIPに入ってきた「PLUCKED．VTD」を使ってください。そのままロードするとBASICを破壊してしまいますので，

LOADM"PLUCKED.VTD",&hB000

としてから聞いてください。琴の音色を選択しているようですが，やはり正統派のピアノでも鳴らしてみましょう。作者の神生君はMML歴1カ月だそうです。皆さんも負けずに投稿してくださいね。　(S.K.)

リスト1　風の谷のナウシカ

```
10 /*              風の谷のナウシカ
20 /*
30 /*              OPMA VERSION
40 /*
50 /*              Music by 細野晴臣
60 /*
70 /*         Programed by 安藤正洋
80 /*
90 /*
100 /*    S O U N D   D A T A
110 /*
120 dim char HHSH(4,10)={ /* ハイハット & シェイカー
130       60, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
140       31,  0,  0,  0,  0,  0,  0, 15,  3,  0,  0,
150       18, 19, 18,  8,  0, 20,  0,  5,  7,  0,  0,
160       31,  0,  0,  0,  0,  0,  0,  8,  7,  3,  0,
170       31, 15, 19,  8,  0, 21,  0,  8,  3,  1,  0}
180 m_vset(70,HHSH)
190 dim char CLAVES(4,10)={ /* クラベス・・・ニハ キコエンカナ コリャー
200       56, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
210       31, 16,  0,  5, 10, 52,  2,  2,  0,  2,  0,
220       31, 12,  0,  5, 10, 30,  2,  0,  0,  0,  0,
230       31, 12,  0,  5, 10, 40,  2,  1,  0,  1,  0,
240       31, 12,  6,  8,  8,  4,  2,  2,  0,  0,  0}
250 m_vset(71,CLAVES)
260 dim char CYMBAL(4,10)={ /* シンバル
270       52, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
280       31,  0,  0,  0,  0,  7,  0, 15,  6,  0,  0,
290       31, 22, 12,  5,  1,  6,  0, 13,  2,  0,  0,
300       31, 25,  0,  2,  0,  8,  0, 10,  2,  0,  0,
310       31, 25, 13,  5,  4,  6,  0, 15,  6,  0,  0}
320 m_vset(72,CYMBAL)
330 dim char SYNBASS(4,10)={ /* シンセベース
340       61, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
350       31, 18,  0, 10,  2, 36,  3,  5,  3,  0,  0,
360       31, 15,  4, 10,  0, 15,  0,  4,  5,  0,  0,
370       31, 15,  4, 10,  0, 25,  0,  2,  7,  0,  0,
380       31, 15,  7, 10,  0,  0,  0,  1,  3,  0,  0}
390 m_vset(73,SYNBASS)
400 dim char STRINGS1(4,10)={ /* バイオリンケイ ノ ストリングス
410       58, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
420       25, 15,  0,  4,  0, 36,  1,  4,  3,  0,  0,
430       23, 15,  0,  5,  0, 34,  1,  8,  0,  0,  0,
440       23, 15,  0,  5,  0, 42,  1, 12,  7,  0,  0,
450       19, 15,  2,  7,  0,  4,  1,  8,  0,  0,  0}
460 m_vset(74,STRINGS1)
470 dim char STRINGS2(4,10)={ /* コントラバスケイ ノ ストリングス
480       58, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
490       24,  6,  0,  5,  1, 32,  2,  2,  3,  0,  0,
500       22, 10,  0,  5,  1, 36,  2, 12,  0,  0,  0,
510       22, 31,  0,  7,  0, 31,  1,  2,  7,  0,  0,
520       16,  3,  0,  7,  0,  0,  1,  2,  0,  0,  0}
530 m_vset(75,STRINGS2)
540 dim char BRASS1(4,10)={ /* トランペット
550       58, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
560       12, 14,  0,  4,  1, 24,  2,  2,  7,  0,  0,
570       13, 14,  0, 10, 15, 38,  2, 14,  0,  0,  0,
580       12, 14,  0,  4,  1, 38,  2,  2,  3,  0,  0,
590       19, 15,  0,  8,  0,  3,  1,  2,  0,  0,  0}
600 m_vset(76,BRASS1)
610 dim char BRASS2(4,10)={ /* トロンボーンケイ ノ ブラス
620       58, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
630       14, 10,  0,  5,  1, 26,  2,  1,  3,  0,  0,
640       24, 13,  3, 10, 15, 31,  1, 10,  0,  0,  0,
650       15, 11,  0,  5,  2, 53,  1,  1,  7,  0,  0,
660       17,  3,  0,  9,  2,  2,  1,  1,  0,  0,  0}
670 m_vset(77,BRASS2)
680 dim char SYN1(4,10)={ /* コト ノ ヨウナ アナログシンセオン
690       60, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
700       26,  9,  1,  6,  4, 35,  0,  8,  7,  0,  0,
710       23, 15,  9,  8,  0,  0,  0,  8,  7,  0,  0,
720       26,  9,  1,  6,  4, 40,  0,  8,  3,  0,  0,
730       23, 15,  9,  8,  0,  0,  0,  8,  3,  0,  0}
740 m_vset(78,SYN1)
750 dim char SYN2(4,10)={ /* アナログストリングス
760       52, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
770       20,  9,  0,  3,  0, 36,  0,  4,  7,  0,  0,
780       14,  9,  3,  5,  0,  3,  0,  4,  3,  0,  0,
790       20,  9,  0,  3,  0, 32,  0,  4,  3,  0,  0,
800       14,  9,  3,  6,  0,  3,  0,  4,  7,  0,  0}
810 m_vset(79,SYN2)
820 dim char SYN3(4,10)={ /* コーラスオルガン
830        4, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
840       24,  0,  0,  3,  0, 30,  0,  8,  3,  0,  0,
850       22,  0,  0,  6,  0,  0,  0,  8,  3,  0,  0,
860       24,  0,  0,  3,  0, 30,  0,  8,  7,  0,  0,
870       22,  0,  0,  6,  0,  0,  0,  8,  7,  0,  0}
880 m_vset(80,SYN3)
890 dim char SYN4(4,10)={ /* シンセベル
900       52, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
910       31, 15,  0,  4,  0, 24,  0,  8,  7,  0,  0,
920       31, 20, 11,  6,  3,  0,  0,  8,  3,  0,  0,
930       20, 15,  0,  4,  0, 23,  0, 12,  3,  0,  0,
940       20, 20, 11,  6,  3,  0,  0,  4,  7,  0,  0}
950 m_vset(81,SYN4)
960 dim char SYN5(4,10)={ /* アナロウシンセ ノ エレピ
970       52, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
980       31, 15,  0,  4,  0, 25,  0,  8,  7,  0,  0,
990       31, 15, 12,  7,  0,  0,  0,  8,  7,  0,  0,
1000      31, 15,  0,  4,  0, 26,  0, 12,  3,  0,  0,
1010      31, 15, 12,  7,  0,  0,  0,  4,  3,  0,  0}
1020 m_vset(82,SYN5)
1030 dim char SYN6(4,10)={ /* オカリナ
1040      54, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
1050      31,  6, 15, 10,  0, 64,  0,  4,  3,  0,  0,
1060      20, 10,  0, 15,  1,  3,  0,  4,  7,  0,  0,
1070      20, 10,  0, 15,  2,  4,  0,  4,  0,  0,  0,
1080      19, 10,  0, 15,  0,  2,  0,  4,  0,  0,  0}
1090 m_vset(83,SYN6)
1100 dim char SYN7(4,10)={ /* トライアングル ノ ヨウナ オト
1110      63, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
1120      31, 20,  0,  3,  2, 30,  2,  6,  0,  3,  0,
1130      31,  8,  0,  4, 15,  0,  2, 15,  0,  1,  0,
1140      31,  8,  0,  4, 15,  4,  2,  2,  3,  0,  0,
1150      31,  8,  0,  4, 15,  4,  2,  2,  7,  0,  0}
1160 m_vset(84,SYN7)
1170 dim char SYN8(4,10)={ /* ボイス コーラス
1180       4, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
1190      20, 15,  0,  4,  0, 31,  0,  8,  7,  0,  0,
1200      18, 15,  0,  7,  0,  0,  0,  8,  7,  0,  0,
1210      20, 15,  0,  4,  0, 31,  0,  8,  3,  0,  0,
1220      18, 15,  0,  7,  0,  0,  0,  8,  3,  0,  0}
1230 m_vset(85,SYN8)
1240 dim char BRASS3(4,10)={ /* オクターブ ユニゾン ノ ブラス
1250      60, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
1260      13, 16,  0,  6,  2, 22,  0,  2,  7,  0,  0,
1270      18, 15,  0,  8,  0,  0,  0,  2,  3,  0,  0,
1280      12, 15,  1,  6,  4, 26,  0,  1,  3,  0,  0,
1290      15, 15,  1,  8,  0,  0,  0,  1,  7,  0,  0}
1300 m_vset(86,BRASS3)
1310 dim char LEAD(4,10)={ /* メロディーライン
1320      60, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
1330      18, 17,  0,  3,  0, 36,  0,  4,  7,  0,  0,
1340      14, 16,  8,  7,  0,  8,  1,  2,  7,  0,  0,
1350      31, 15,  0,  3, 10, 20,  2, 12,  3,  0,  0,
1360      28, 15,  6,  7,  0, 10,  2,  2,  3,  0,  0}
1370 m_vset(87,LEAD)
1380 /*
1390 /*     初期設定
1400 /*
1410 m_init()
1420 for i=1 to 8:m_alloc(i,5000):m_assign(i,i):next
1430 str a(11)[256],d(1)[256]
1440 m_tempo(112)
1450 /*
1460 /*     M M L  &  m _ t r k ( )
1470 /*
1480 a(0)="@87q818o4v11y48,20 |:6r1:|
1490 a(1)="|:dddee2dd4ee2f+4.e4d|1>a2.r2:||2<a2.r2
1500 a(2)="|:>a<ddd4d4dd4.c+|1c+4c+>ba2.r4r1<:||2c+c+4>ba2.r4r1
<
1510 a(3)="v12f+f+f+f+f+e4.bf+4ed2def+def+ef+ed&d2r4
1520 a(4)="f+f+f+f+f+a+4.bf+4ed2def+def+ef+ed4dedc+df+2.r4v13f+
4e4a4g4gf+2..r4.v12d&edc+dv11f+2e2d2c+2
1530 a(5)="d2.r4r1
1540 a(6)="r1r1
1550 for i=0 to 2:m_trk(1,a(i)):next
1560 for i=1 to 6:m_trk(1,a(i)):next
1570 for i=1 to 5:m_trk(1,a(i)):next
1580 for i=4 to 5:m_trk(1,a(i)):next
1590 /*
1600 /*
1610 a(0)="@74q818o4v10 c4&>{b&b-}a4&{a-&g}f+4&{e&e-}d2..{c+>ba
b<c+dedc+def+gf+ef+gabagab<c+}1d2e2
1620 a(1)="@78o2v9 d4f4e4c4d4f4{efe}dc4
1630 a(2)="@85o2v5 |:d>ab<c+d4a4d4>b4<c+4a4f+4>b4<e4d4d4>b4<c+4
>a4<:|
1640 a(3)="@83o3 |:r{v3a<v6dv8f+}v9q4a{ab}q8a1&a2|1v2d{v4ev6f+v
8g}v9q4aaaq8a4a4q4gf+q8e4.r4:||2
1650 a(4)="r4<{dc+}d4q4>bagq8g16f+8.e2.
1660 a(5)="@74o3v10 r{ab<c+dc+>b}4r4{ab<c+dc+>b}4r4{ab<c+dc+d}4
{ef+g}l16araaeearl8
1670 a(6)="@74o4v7y50,40 l16f+2r8q4gf+f+c+gf+q8f+2r8q4gf+f+>a<g
f+q8f+2e2d2v11{dc+>bv10agf+gab<c+de}2l8 y51,0
1680 a(7)="@74o4v7y50,40 l16f+2r8q4gf+f+c+gf+q8f+2r8q4gf+f+>a<g
f+q8f+2e2dl18 y51,0
1690 a(8)="v8d1d2c+2c+2>af+daf+1>v7b1b2b-2
1700 a(9)=a(1)+a(1)
1710 a(10)="@74o4v10 c4&>{b&b-}a4&{a-&g}f+4&{e&e-}d4.>{v11ab<c+
v10def+>v11b<c+def+g}2
1720 a(11)="@74o4v10 c4&>{b&b-}a4&{a-&g}f+4&{e&e-}d4.v9c+2<{v7c
+&v8c+&v9c+}2&c+1&{v6c+&v3c+&v0c+}4
1730 for i=0 to 1:m_trk(2,a(i)):next
1740 m_trk(2,"|:8r1:|")
1750 for i=3 to 4:m_trk(2,a(i)):next
1760 for i=2 to 3:m_trk(2,a(i)):next
1770 for i=5 to 9:m_trk(2,a(i)):next
1780 for i=2 to 3:m_trk(2,a(i)):next
1790 for i=5 to 8:m_trk(2,a(i)):next
1800 m_trk(2,a(10))
1810 for i=7 to 8:m_trk(2,a(i)):next
1820 m_trk(2,a(11))
1830 /*
1840 /*
1850 a(0)="@86q818o4v8y50,40 c4.>a4.f+4.d2..f+1<d2e2 y50,0
1860 a(1)="r1r1
1870 a(2)="|:7r1:|r2 @81o3v11 ec+>ag
1880 a(3)="@74o1v7 |:3a2b2<d2c+2>:|
1890 a(4)="a2b2<d2c+2
1900 a(5)="@75o2v8 a4.{ab<c+dc+>b}4b4{ab<c+}{dc+>bab<c+dc+def+g
}2l16araaeearl8
1910 a(8)="@84o4v7 f+f+4f+dd4de2r2ee4ec+c+4c+>a2r2|:4ra<a4>:|
1920 a(9)="@77o4v9 l16d8>aaa8<@76aa@77d8>aaa8<dd|:d8>aaa8<@76aa
a8@77>aaa8<dd:|d8>aaa8<@76aaa8{v5a&v7a&v9a&v10a}8&v11a4l8
1930 a(10)="@86o4v8y50,40 c4.>a4.f+4.d4.a4<c+4 y50,0
1940 a(11)="@86o4v8y50,40 c4.>a4.f+4.d4.v7c+2<{v6c+&v7c+&v8c+}2
&c+1&{v6c+&v3c+&v0c+}4 y50,0
1950 for i=0 to 4:m_trk(3,a(i)):next
1960 for i=2 to 3:m_trk(3,a(i)):next
1970 for i=5 to 9:m_trk(3,a(i)):next
1980 for i=2 to 3:m_trk(3,a(i)):next
1990 for i=5 to 8:m_trk(3,a(i)):next
2000 m_trk(3,a(10))
2010 for i=7 to 8:m_trk(3,a(i)):next
2020 m_trk(3,a(11))
2030 /*
2040 /*
2050 a(0)="@76q818o4v8 c4.>a4.f+4.d2..f+1<d2e2
2060 a(1)="@79o2v4 d1d1
2070 a(2)="@79o2v4 |:d1>b2b2g2e2<d2c+2:|
2080 a(3)="@82o3v5 |:3f+2g4f+4e2e2:|
2090 a(4)="f+2g4f+4e2e2
2100 a(5)="f+2g4f+4e2r @76o4v10 l16aaeearl8
2110 a(6)="@79o1v4 b2b-2b2<c2>b2b-2b2b4a4
2120 a(7)="@79o1v4 b2b-2b2<c2>b2b-2a1
2130 a(8)="@86o4 {v3f+&v4f+&v5f+&v6f+&v7f+}1a1f+2v8<af+daf+1 @7
9o2v4 d1>b1
2140 a(9)=a(1)+"d1 @80o3 {v0d&v1d&v2d&v3d&v4d}2&v5d2
2150 a(10)="@76o4v8 c4.>a4.f+4.d4.e4a4
2160 a(11)="@76o4v8 c4.>a4.f+4.d4. @79o2v4c+2<c+1&{v3c+&v2c+&v1
c+&v0c+}4
2170 for i=0 to 4:m_trk(4,a(i)):next
2180 for i=2 to 3:m_trk(4,a(i)):next
2190 for i=5 to 9:m_trk(4,a(i)):next
```


▶ゲームボーイLIFE (Vol.2) の“試験に出ないゲームボーイ講座”とは祝氏に対する挑戦か？
吉田俊一(19)山形県

```
2200 for i=2 to 3:m_trk(4,a(i)):next
2210 for i=5 to 8:m_trk(4,a(i)):next
2220 m_trk(4,a(10))
2230 for i=7 to 8:m_trk(4,a(i)):next
2240 m_trk(4,a(11))
2250 /*
2260 /*
2270 a(0)="@77q818o4v8 c4.>a4.f+4.d2..f+1<d2e2
2280 a(1)="@79o1v4 a1a1
2290 a(2)="@79o1v4 |:a1f+2f+2d2>b2<b2a2:|
2300 a(3)="@82o3v5 |:3d2d4d4d2c+2:|
2310 a(4)="d2d4d4d2c+2
2320 a(5)="d2d4d4d2r @86o3v8 l16aaeearl8
2330 a(6)="@79o1v4 g2f+2f+2a2g2f+2f+2g4e4
2340 a(7)="@79o1v4 g2f+2f+2a2g2f+2f1
2350 a(8)="@80o3v5 f+1c+1f+1d1 @79o1v4 b1g1
2360 a(9)=a(1)+"a1 @80o3 {v0a&v1a&v2a&v3a&v4a}2&v5a2
2370 a(10)="@77o4v8 c4.>a4.f+4.d4.e4a4
2380 a(11)="@77o4v8 c4.>a4.f+4.d4. @79o1v4 a2<a1&{v3a&v2a&v1a&v
0a}4
2390 for i=0 to 4:m_trk(5,a(i)):next
2400 for i=2 to 3:m_trk(5,a(i)):next
2410 for i=5 to 9:m_trk(5,a(i)):next
2420 for i=2 to 3:m_trk(5,a(i)):next
2430 for i=5 to 8:m_trk(5,a(i)):next
2440 m_trk(5,a(10))
2450 for i=7 to 8:m_trk(5,a(i)):next
2460 m_trk(5,a(11))
2470 /*
2480 /*
2490 a(0)="@75q818o2v9 d4.>b4.g4.e2..a1a2a2
2500 a(1)="@79o1v4 f1f1
2510 a(2)="@79o1v4 |:f1d2e2>b2g2<g2e2:|
2520 a(3)="@82o3v5 >|:3a2b4a4a2a2:|
2530 a(4)="a2b4a4a2a2
2540 a(5)="a2b4a4a2r2
2550 a(6)="@79o1v4 f+2c+2d2f+2f+2c+2d2e4c+4
2560 a(7)="@79o1v4 f+2c+2d2f+2f+2c+2c1
2570 a(8)="@80o3v5 e1>a1<d1>b1< @79o1v4 e1e1
2580 a(9)=a(1)+"f1 @80o3 {v0f&v1f&v2f&v3f&v4f}2&v5f2
2590 a(10)="@75o2v9y53,20 d4.>b4.g4.e4.<c+4c+4 y53,0
2600 a(11)="@75o2v9y53,20 d4.>b4.g4.e4.v8a2<{v6d&v7d&v8d}2&d1&{
v6d&v3d&v0d}4 y53,0
2610 for i=0 to 4:m_trk(6,a(i)):next
2620 for i=2 to 3:m_trk(6,a(i)):next
2630 for i=5 to 9:m_trk(6,a(i)):next
2640 for i=2 to 3:m_trk(6,a(i)):next
2650 for i=5 to 8:m_trk(6,a(i)):next
2660 m_trk(6,a(10))
2670 for i=7 to 8:m_trk(6,a(i)):next
2680 m_trk(6,a(11))
2690 /*
2700 /*
2710 a(0)="@73q418o1v12 |:4r1:|
2720 a(1)="|:8d<d>:|
2730 a(2)="|:d<d>d<d>d<d>d<d>>b<b>b<b>b<b>b<b>g<g>g<g>e<e>e<e>a
<a>a<a>a<a|1>a<a:||2>g<g
2740 a(3)=">|:3f+<f+>f+<f+>g<g>g<g>a<a>a<a>g<g>g<g>:|
2750 a(4)="f+<f+>f+<f+>g<g>g<g>a<a>a<a>a<a>a<a
2760 a(5)="f+<f+>f+<f+>g<g>g<g>a<a>a<ar{eeee}4e
2770 a(6)=">|:g<g>g<g>f+<f+>f+<f+>b<b>b<b>|1a<a>a<a>:||2e<e>a<a
2780 a(7)=">|:g<g>g<g>f+<f+>f+<f+>|1b<b>b<b>a<a>a<a>:||2|:4f<f:
|
2790 a(8)=">|:4e<e>:||:4a<a>:|<|:4d<d>:|>|:4b<b>:||:4e<e>:||:4a
<a>:|<
2800 a(9)=a(1)+a(1)
2810 a(10)="q8d4.>b4.f+4.e4.<q4c+c+c+c+
2820 a(11)="q8d4.>b4.f+4.e4.a2<v11d1&d2.
2830 for i=0 to 4:m_trk(7,a(i)):next
2840 for i=2 to 3:m_trk(7,a(i)):next
2850 for i=5 to 9:m_trk(7,a(i)):next
2860 for i=2 to 3:m_trk(7,a(i)):next
2870 for i=5 to 8:m_trk(7,a(i)):next
2880 m_trk(7,a(10))
2890 for i=7 to 8:m_trk(7,a(i)):next
2900 m_trk(7,a(11))
2910 /*
2920 /*
2930 d(0)="y2,2r8@70ggy2,10r8ggy2,2r8ggy2,10@71g8@70gg
2940 d(1)="y2,2r8@70ggy2,10r8ggy2,2r8y2,2ggy2,10@71g8@70gg
2950 a(0)="@11r q8l16o4 @72v15 <g4.g4.g4.g2..v8|:{gggggg}4|1v9:
||2v10:||3v11:|v15g2g2
2960 a(1)="y3,3|:3y2,2r8@70gg:|y2,2@71g8@70gg|:2y2,2r8@70gg:|y2
,2r8y2,10@70ggy2,10@71g&y2,10gy2,2@70gg
2970 a(2)="|:3"+d(0)+":|
2980 a(3)="|:5"+d(1)+":|"+d(0)+"|:5"+d(1)+":|"+d(0)
2990 a(4)="y2,2r8@70ggy2,10r8ggy2,2r8y2,2ggy2,10@71g&y2,10gy2,2
@70gg
3000 a(5)="y2,2r8@70ggy2,10r8ggy2,2r8y2,10gy2,10gy2,10@71g&y2,1
0gy2,10@70gg
3010 a(6)="|:4"+d(0)+":|
3020 a(7)="|:10"+d(0)+":|
3030 a(8)="|:3"+d(0)+":|"+"y2,2r8@70ggy2,10r8ggy2,2r8y2,10ggy2,
10@71g&y2,10gy2,2@70gg
3040 a(9)=" y2,2@72<g4&y2,10g8y2,2g4.y2,2g4&y2,10g8y2,2g8&y2,2g
4y2,2r8y2,10>@70ggy2,10@71g8y2,10@70gy2,10g
3050 a(10)="y2,2@72<g4&y2,10g8y2,2g4.y2,2g4&y2,10g8y2,2g4.y2,2r
8>@70gg@71y2,2g8@70gg
3060 for i=0 to 4:m_trk(8,a(i)):next
3070 for i=2 to 3:m_trk(8,a(i)):next
3080 for i=5 to 8:m_trk(8,a(i)):next
3090 for i=2 to 3:m_trk(8,a(i)):next
3100 for i=5 to 7:m_trk(8,a(i)):next
3110 m_trk(8,a(9))
3120 m_trk(8,a(7))
3130 m_trk(8,a(10))
3140 /*
3150 /*     P L A Y
3160 /*
3170 m_play()
```

日本音楽著作権協会(出)許諾第9071006-001号

リスト2　ラジオ体操第一

```
10 '  ﾗｼﾞｵ ﾀｲｿｳ No.1
20 '
30 '            1990.6.21.  by N.Kamio
40 '
50 '
60 TEMPO 0
70 PLAY "t110"
80 PLAY " i40v13p3q7=3s2,8,0,5";
90 PLAY ":i40v13p1q7=3s2,8,0,5";
100 PLAY ":i40v13p2q7=3s2,8,0,5";
110 PLAY ":i40v10p3q7=3s2,8,0,5";
120 PLAY ":i40v10p2q7=3s2,8,0,5";
130 PLAY ":i40v10p1q7=3s2,8,0,5"
140 '----------------  1  ------------------------------------
150 PLAY "o5l8d4f+.g16adf+ag4b.o6c+16do5gbo6do5ao6do5bagf+det100
f+dec+d4t70o4f+4";
160 PLAY ":o4l8a4o5d.e16r2d4g.a16br4.r1t100r2.t70r64o4a@45";
170 PLAY ":o4l8f+4r2.b4r2.r1t100r2.t70r32o5d@45";
180 PLAY ":o3l4dao4dr4o2go4dgr4ao4f+o3bo4dt100o2ao3af+t70r32.o5f
+@39";
190 PLAY ":r1r1o3l4f+o4do3gbt100ro3gdt70r16o5a@36";
200 PLAY ":r1r1o3l4dbegt100r2.t70r16r64o6d@33"
210 '----------------  2  ------------------------------------
220 PLAY "t70o4l4rf+af+rf+af+o5!125f+l16f+8.ga8.b32a32f+8.do5!12
5b4b8.o6c+!125d4.c+o5b!125a4a8.ab8.ad8.e";
230 PLAY ":o4@l45r4r64dr64f+r64dr4r64dr64f+r64dl4r1r1r1";
240 PLAY ":o3@l42r4r32ar32o4dr32o3ar4r32ar32o4dr32o3al4r1r1r1";
250 PLAY ":o3l4dro2aro3dro2aro3dadadbdbdada";
260 PLAY ":o2l4dr2.dr2.rf+rf+rgrgrf+rf+";
270 PLAY ":l1rrrrr"
280 '----------------  3  ------------------------------------
290 PLAY "o5l16!125f+8.ga8.f+!125e4o4a4o5!125f+4f+8.ga8.b32a32f+
8.d!125b4b8.o6c+!125d4.c+o5b!125a4a8.o6do5b8.ad8.e!125f+8.de8.c+
!125d4o6d4";
300 PLAY ":l1rrrrr2.r64o5a@45";
310 PLAY ":l1rrrrr2.r32o5f+@42";
320 PLAY ":o2l4ao3ao2ao3adadadbdbdao2go3bo2ao3aad";
330 PLAY ":o3l4rf+rgrf+rf+rgrgrf+rgrgf+o2d";
340 PLAY ":o3l4rdrc+r1r1r1r2dr"
350 '----------------  4  ------------------------------------
360 PLAY "o5l16e4e8.d!125c+8.o4aa8.a16o5d8.c+d8.e!125f+2f+4f+8.e
!125d+8.o4bb8.bo5e8.d+e8.f+!125g2o4!125a4f+8.ao5!125d8.o4aa4";
370 PLAY ":o5l8r64c+@45r2.r1r64d+@45r2.r1o4!125f+4d.f+16!125a.f+
16f+4";
380 PLAY ":o4l2r32a@42r.r1r32b@42r.r1!125d4r.";
390 PLAY ":o2l4ao3a!114ao4c+o3da!114f+a+o2bo3f+o2!114bo3f+o2eo3g
o2!114ao3gdada";
400 PLAY ":o3l4reo2!114ao3arf+o2!114f+o3f+rd+o1!114bo3dreo1!114a
o3eo2do3f+o2do3f+";
410 PLAY ":o3l4r2.er1r1ro2bro3c+rdrd"
420 '----------------  5  ------------------------------------
430 PLAY "o4l8!125b4g.b16o5!125e.o4b16b4o5!125d4o4a.b16o5!125c+.
d16f+.e16!125d4o4a.a16o5!125d4d4o4d4.ed4o3b4o4g.a16b4b2";
440 PLAY ":o4l8!125g4e.g16!125b.g16g4!125a4f+.g16!125a.b16o5d.c+
16o4!125a4a.a16!125a4a4r1r4gg2";
450 PLAY ":o4l8!125e4r2.!125f+4r2.!125f+4r4!125f+4f+4r1r2d2";
460 PLAY ":o3l4ebebdao2ao3gdaaddddddddd";
470 PLAY ":o2l4eo3eo2eo3go2do3f+reo2do3gf+o2dbbbbbdgg";
480 PLAY ":o3l4rererdrc+rc+dro2gggggr2."
490 '----------------  6  ------------------------------------
500 PLAY "o4l8!125e.g16o5c4o4b4a4g.e16b4b2o5e4g4f+.g32f+32e4v15d
4g4d.v13e32d32o4!125b4a4o5e4d.e32d32c4";
510 PLAY ":o4l4r1red+2o5cedcv15o4bo5do4bgv13eo5co4ba";
520 PLAY ":o3l1rrrrr";
530 PLAY ":o4l4cco3babgf+2cgcgo2go3go2go3gcaca";
540 PLAY ":o3l4ggggeed+2rererdrdrere";
550 PLAY ":o3l4eecco2bbb2r1r1r1"
560 '----------------  7  ------------------------------------
570 PLAY "v15o4l8b4a4o5d2drd.!119e16dro4bro5drg.!119b16o6dro5bro
6crerd.!119c16o5b.!119a16";
580 PLAY ":v15o4l1f+4f+4a2rrr";
590 PLAY ":v15o4l1r2f+2rrr";
600 PLAY ":o2l4do3f+dal8gro4dro3dro4dro3gro4dro3dro4dro3crgrdrar
";
610 PLAY ":o3l4r4do2do3f+r4b8r.b8r.b8r.b8r.e8r.f+8r8";
620 PLAY ":o3l4rrrdr1r1r1"
630 '----------------  8  ------------------------------------
640 PLAY "o5l8grbrgro6g4o4l4ao5dv13f+8.e16d8r8v15dgv13b8.a16g8r8
v15f+av13f+d8.e16";
650 PLAY ":o6l4r2.do4f+ar2gbr2o5df+v13dr";
660 PLAY ":o5l4r2.bo4df+r2gbr2ao5dr2";
670 PLAY ":o4l8drdrdro2g4l4dadado4do3do4do3dadb";
680 PLAY ":o3l8brdrbr4.l4o2do3f+rf+o2do3brbo2do3f+rf+";
690 PLAY ":o3l8gr4.gr4.l4ro3drdrgrgrdr2"
700 '----------------  9  ------------------------------------
710 PLAY "v15o5l4f+f+v13f+el16r16o3ag+abo4c+def+8.ed8r8.o3ba+bo4
c+def+g8.f+e8r8.dc+def+gab8.ag8r8";
720 PLAY ":v15o5l4ddv13c+c+r1r1r1";
730 PLAY ":o4l4aav13aar1r1r1";
740 PLAY ":o2l4ao3ao2ao3a!117dddd8r8!117eeee8r8!117ddee8r8";
750 PLAY ":o1l4ao3f+rgo2!117aaaa8r8!117bbbb8r8!117aabb8r8";
760 PLAY ":o3l4rdr2o2!117dddd8r8!117eeee8r8!117ddee8r8"
770 '----------------  10 ------------------------------------
780 PLAY "o4l16r16ag+bagf+ed8.o3ao4d8r8t100o5l8q5f+4f+ga.q8b32a3
2q6f+db4bo6c+d4.q8c+16o5b16q6a4aabade";
790 PLAY ":l1rt100rrr";
800 PLAY ":l1rt100rrr";
810 PLAY ":o3l4c+c+dd8r8t100l8q5daf+adaf+adbgbdbgbdaf+adaf+a";
820 PLAY ":o2l4aaaa8r8t100r1r1r1";
830 PLAY ":o2l4c+c+dd8r8t100r1r1r1"
840 '----------------  11  ------------------------------------
850 PLAY "o5l8f+gaf+t70q7v15e4o4a4o5f+4v13f+.g16!125a.b32a32f+.d
16!125b4b.o6c+16!125d4.c+16o5b16!125a4a.o6d16o5!125b.a16d.e16";
860 PLAY ":l1r2t70r2rrr";
870 PLAY ":l1r2t70r2rrr";
880 PLAY ":o2l8ao3af+ao2t70q7a4o3g4l4dadadbdbdadb";
890 PLAY ":o3l4r2t70rerf+rf+rgrgrf+r4g";
900 PLAY ":o3l1r2t70r4c+4rrr"
910 '----------------  12  ------------------------------------
920 PLAY "o5l4!125f+8.d16!125e8.c+16do6do4bo5d8.c+16do4baf+a2bo5
d8.c+16do4bt65o5dt60eq8t50f+\t40o6d2v0";
930 PLAY ":o5l4r2.r64a@45o4gb8.a+16bgf+df+2gb8.a+16bgt65at60o5c+
q8t50d\t40r64a@45&av0";
940 PLAY ":o5l4r2.r32f+@42r1r1r1t65o4f+t60gq8t50a t40r32o5f+@42&
f+v0";
950 PLAY ":o2l4ao3aado2go3dgbo2dao3df+o2go3dgbt65dt60aq8t50d t40
o2d2v0";
960 PLAY ":o3l4rgf+o2dr1r1r1t65rt60rt50rt40r2v0";
970 PLAY "o3l1r2d4r4rrrl4t65rt60rt50rt40r2v0"
980 END
```

▶通巻100号，おめでとうございます。"表紙ぎゃらりぃ"を見て100号の長さと歴史を感じました。いやぁ，本当にすごい。
山上秀景(17)東京都

★(で)のショートプロぱーてぃ その13

なさけなくない星？

Komura Satoshi 古村　聡

ぱーてぃハンズのほうも今月でとりあえず第1部「完」ということになり，めでたしめでたし。それを記念して(?)今月も2本のプログラムを用意してみました。X1用ツール「Date Changer」とX68000用のデモ「なさけない★星★」です。

illustration : T.Takahashi

皆様，夏休みいかがお過ごしで？　まあ，この時期になりますってーと毎年書いてるような気もするんですがね。休みってなぁいいもんですなあー。夏休みの目標「今年の夏休みは大作のプログラムを作るぞ！」なんて思わず力んでしまいます。

で，朝マジメに起きて6:30からラジオ体操行って，午前中にプログラム組んで夜は9:00に寝ちゃうなんていうオカタイ方には関係ないでしょうが，朝は昼まで寝てて（どういう日本語じゃ）プログラミングを午前3時までなんていう私のような完璧夜型人間にとりまして，プログラムを組むときいちばん気になるのはやっぱり「その日の夜食が何であるか」なんですなこれが（こら，そこ！　おめーだけだよ，なんて言わないよーに)。私のお気に入りの夜食はっつーとですねえ。えへへ，当ててごらんなさい。何でしょう。え，ポテトフライ？　歌舞伎揚げ？　毒物飲料の一気飲み？　あんたね，なにも自殺しようってんじゃないんだから……。へへへへ，実は「鯨の焼肉」だったりするんですなこれが。

あれは3年前，捕鯨が禁止されることになって，もう鯨の焼肉の缶詰が食べられなくなると聞いた私はそこら中のスーパーやら酒屋やらを走り回って鯨の缶詰を30缶ほど買いあさったんですね。んで，3年たったいまでも20缶ほど残っていてそれをたまに「いまでもこんなもん食べてるの世界中でわしひとりだろーなー。えへへへへ」と優越感（優越缶？）に浸りつつ食しているというわけです。

えっ，なんでこんな話をするのかって？

単なる自慢話に決まってるじゃあないですか。えへへへへ。

日付よ,かわれ!

さて，そんな楽しいプログラミングの役に立つ……かどうかはわかりませんが，ちょっと便利なユーティリティ。今月の1本目は北海道の赤木さんの作品でX1turbo用のDate Changerです。

Date Changer for X1turboシリーズ
(CZ−8FB02)

北海道　赤木陽一郎

このツールはturboBASIC用です，注意してください。たぶん，画面関係をいじくればX1 BASICでも動くのではないか，と作者の赤木さんも言っていますからX1で動かしたい方はぜひそのようにしてください。私はやりませんでした(漢字が絡んだらX1だとめんどくさそうだなとか考えてしまったのだ。が，よく考えたらグラフィックキャラクタが出てくるだけなんだよね)。

で，このプログラムでなにができるのか？　読んで字のごとく，ファイルの日付や曜日，時間を変えるんです。BASICでFILESをとると，たとえば，

Bin＊“ 0:BASIC CZ8FB02.Sys” '85/10/15 TUE 12:00

なんて出てきますよね。こいつの

'85/10/15 TUE 12:00

の部分を変えられるプログラムなんです。ま，早い話がHuman68kでいうところの「touch.x」なんですね（しまった,X1ユーザーにはますますわからんじゃないか)。

では，使い方を。まず，turboBASICを立ち上げます。で，このプログラムのリストを打ち込んでください。で，プログラムをセーブ。うん，ここまではいつもと一緒ですね。で，今回はプログラムが直接ディスクをいじりにいってしまうので，もし打ち間違いがあると非常に危険なんです（最悪の場合，そのディスクからLOAD，SAVEが一切できなくなってしまいます)。そこで，最初はディスクを壊してもいいようにFormat＆Copyを使ってディスクのバックアップを取ってください。ま，壊してもいいようなディスクがあれば，それで試してみてもいいですが。とりあえず必要なディスクならちゃんと別に取っておくのです。で，RUN。[1]で日付，曜日，年の変更，[2]でひとつのファイルの日付のクリア，[3]でオールクリアです。とりあえずなにかキーを押してみれば，なにか向こうからいってくるのでわかるのではないかと思います。そうそう，4のエンドを選択しないと日付が書き込まれませんのでご注意を。そして，くれぐれも打ち込みのときは間違わないよう注意してください。

これってFATとかディレクトリとかがわかるようになると一度は作ってみたくなるプログラムですね。私もFATとディレクトリの存在を知ったときには，「ファイルの内容をダンプするツール」を作ったお

なさけない★星★

ぼえがあります。ちなみにそのときは1回目でシステムディスクをぶっ壊しました。うーん，なつかしい思い出だなぁ。そういえば，最近はあまりFATとかの解説とかスキューフォーマッタとかってあまり見ないなあ……。まあ，今となってはあまり憶える必要のある概念じゃないかもしれないけどねー。そういや，マシン語のプログラムのテストランじゃディスクを抜かなきゃいけないなんて知らない人も多いのかもしれないなー。昭和は遠くなりにけり。

星に願いを……

では今月の2本目。今月の2本目は埼玉県の遠藤さんの作品でX68000用のデモプログラム，「なさけない★星★」です。

なさけない★星★ for X68000 (C compiler PRO—68K)

埼玉県　遠藤克之

Cコンパイラを使って作られたデモプログラムです。このプログラムを実行するためにはXC(C compiler PRO-68K)が必要です。

まず,ED.Xなどのエディタを使って打ち込みます。コマンドモードで(面倒でもコマンドモードで立ち上がるディスクを作っておいてください。ま，メモリに余裕があればVSやSX-WINDOWからCOMMAND.Xを立ち上げてもかまいませんけどね……)，

1)　ed HOSI.C

としてエディタを立ち上げてリストを打ち込んでください。打ち込み終わったらこのリスト（このリストのことをソースといいます）をコンパイルして実行形式（.Xファイル）にします。

2)　cc /O /Y /W HOSI.C

これで何もエラーメッセージが出なければOKです。エラーが出てしまった場合は1)

リスト1　Date Changer

```
10 '
20 '        D a t e   C h a n g e r         Ver. 1.0
30 '
40 '                                       (C)Yoichiro
50 '
60 ' ///////// SHOKI /////////
70 WIDTH 80,25:KLIST 0:INIT:CLS 4:CONSOLE 0,25:DEFINT A-Z
80 MAX=1:RN=16:DIM FA$(60),DA$(60,4),FI$(60)
90 FOR I=0 TO 6:READ DY$(I):NEXT I
100 DATA SUN,MON,TUE,WED,THU,FRI,SAT
110 LOCATE 12,0:COLOR 3:PRINT "＜＜＜";SPC(44);"＞＞＞";:COLOR 6
:LOCATE 22,0:PRINT "Ｄａｔｅ　Ｃｈａｎｇｅｒ";:COLOR 7:PRINT "
 Ver. 1.0"
120 CONSOLE 1,3,10,70:CLS:COLOR 7:PRINT:INPUT "DRIVE No.=",DEV$
125                 CLS:COLOR 7:PRINT:PRINT "Ｆｉｌｅ　読み込
み中　！！";
130 DEVI$ DEV$,RN,X$,Y$
140 FOR I=0 TO 3:DT$=MID$(X$,I*32+1,32):GOSUB 630:NEXT I
150 FOR I=0 TO 3:DT$=MID$(Y$,I*32+1,32):GOSUB 630:NEXT I
160 RN=RN+1:GOTO 130
170 MAX=MAX-1
180 ' ///////// MAIN /////////
190 GOSUB 710
200 COLOR 7:CONSOLE 2,2,4,76:INPUT "1..DATE_CHANGE  2..ONE_CLEAR
  3..ALL_CLEAR  4..END  SELECT(1-2)";A:IF A<1 OR A>4 THEN 200
210 ON A GOTO 220,400,450,510
220 ' ///////// DATE_CHANGE /////////
230 CLS:PRINT "DATE CHANGE   SELECT(No.1-No.";MAX;")";:INPUT A:I
F A<1 OR A>MAX THEN 230
240 CONSOLE 6,15,9,62:CLS:LOCATE 12,7:PRINT "FILENAME ";DEV$;FA$
(A);"  "
250 PRINT "   DATE&TIME  ";DA$(A,0);"/";RIGHT$("0"+RIGHT$(STR$(V
AL("&H"+LEFT$(DA$(A,1),1))),LEN(STR$(VAL("&H"+LEFT$(DA$(A,1),1))
))-1),2);"/";RIGHT$("0"+DA$(A,2),2);" ";
260 PRINT DY$(VAL("&H"+RIGHT$(DA$(A,1),1)));" ";RIGHT$("0"+DA$(A
,3),2);":";RIGHT$("0"+DA$(A,4),2)
270 LOCATE 16,10:INPUT "Input Year   ";B$:B=VAL(B$):IF B<0 OR B>
99 OR LEN(B$)<>2 THEN 270
280 LOCATE 16,11:INPUT "Input Month  ";C$:C=VAL(C$):IF C<0 OR C>
12 OR LEN(C$)<>2 THEN 280
290 LOCATE 16,12:INPUT "Input Day    ";D$:D=VAL(D$):IF D<0 OR D>
31 OR LEN(C$)<>2 THEN 290
300 LOCATE 16,13:INPUT "Input Week   ";E$:E=-1:FOR I=0 TO 6:IF D
Y$(I)=E$ THEN E=I
310 NEXT I:IF E=-1 THEN 300 ELSE E$=RIGHT$(STR$(E),1)
320 LOCATE 16,14:INPUT "Input Hour   ";F$:F=VAL(F$):IF F<0 OR F>
12 OR LEN(F$)<>2 THEN 320
330 LOCATE 16,15:INPUT "Input Minute ";G$:G=VAL(G$):IF G<0 OR G>
59 OR LEN(G$)<>2 THEN 330
340 LOCATE 18,17:COLOR 4:PRINT B$;"/";RIGHT$("0"+RIGHT$(STR$(C),
LEN(STR$(C))-1),2);"/";D$;" ";DY$(E);" ";F$;":"G$
350 LOCATE 20,18:COLOR 7:INPUT "Are you right (Y/N)";A$:IF A$="Y
" OR A$="y" THEN 370
360 IF A$<>"N" AND A$<>"n" THEN 350 ELSE 390
370 DD$(0)=B$:DD$(1)=HEX$(C)+HEX$(E):DD$(2)=D$:DD$(3)=F$:DD$(4)=
G$
380 FOR I=0 TO 4:DA$(A,I)=DD$(I):NEXT I
390 CLS:GOTO 190
400 ' ///////// ONE_CLEAR /////////
410 CLS:PRINT "Input number (1-";MAX;")  ";:INPUT A
420 IF A<1 OR A>MAX THEN 400
430 FOR I=0 TO 4:DA$(A,I)="00":NEXT I
440 CLS:PRINT "Ok.":PAUSE 10:GOTO 190
450 ' ///////// ALL_CLEAR /////////
460 CLS:INPUT "Are you sure (Y/N) ";A$:IF A$="Y" OR A$="y" THEN
480
470 IF A$<>"N" AND A$<>"n" THEN 460 ELSE 190
480 FOR I=1 TO MAX:FOR J=0 TO 4
490 DA$(I,J)="00":NEXT J:NEXT I
500 CLS:PRINT "Ok.":PAUSE 20:GOTO 190
510 ' ///////// END /////////
520 CLS:INPUT "Are you sure (Y/N) ";A$:IF A$="Y" OR A$="y" THEN
540
530 IF A$<>"N" AND A$<>"n" THEN 520 ELSE GOTO 190
540 CLS:PRINT "Ｆｉｌｅ　書き込み中　！！"
550 FOR I=1 TO MAX:FOR J=0 TO 4:MID$(FI$(I),25+J,1)=CHR$(VAL("&H
"+DA$(I,J))):NEXT J:NEXT I
560 FOR I=0 TO MAX¥8:X$="":Y$="":FOR J=0 TO 3
570 X$=X$+FI$(I*8+J+1):NEXT J:FOR J=4 TO 7
580 Y$=Y$+FI$(I*8+J+1):NEXT J
590 X$=X$+STRING$(128-LEN(X$),&HFF):Y$=Y$+STRING$(128-LEN(Y$),&H
FF)
600 DEVO$ DEV$,16+I,X$,Y$
610 NEXT I
620 CONSOLE 0,24:KLIST 1:CLS:KEY0,"FILES"+CHR$(&H22)+DEV$+CHR$(&
H22)+CHR$(13):END
630 ' ///////// FILES READ /////////
640 IF ASC(LEFT$(DT$,1))=&HFF THEN RETURN 170
650 IF ASC(LEFT$(DT$,1))=&H0 OR MAX>60 THEN RETURN
660 FI$(MAX)=DT$:FA$(MAX)=MID$(DT$,2,13)+"."+MID$(DT$,15,3)
670 FOR J=0 TO 4
680 DA$(MAX,J)=HEX$(ASC(MID$(DT$,25+J,1)))
690 NEXT J
700 MAX=MAX+1:RETURN
710 ' ///////// FILES PRINT /////////
720 COLOR 5:CONSOLE 4,21,0,25:CLS:FOR I=1 TO 20:PRINT " No.";RIG
HT$("0"+STR$(I),2);" ";FA$(I):NEXT I
730 CONSOLE 4,21,26,25:CLS:FOR I=21 TO 40:PRINT "No.";I;FA$(I):N
EXT I
740 CONSOLE 4,21,52,25:CLS:FOR I=41 TO 60:PRINT "No.";I;FA$(I):N
EXT I
750 RETURN
```

に戻ってエラーの出たところを直してください。そのときエラーの出たのが何行目か憶えておいて，エディタに入ってから，

[esc]＋行番号＋リターン

とすれば，その行に飛んでくれます。

さて，これでディスク上にHOSI.Xができましたね。それでは電気を消してください（まさか，昼間に打ち込んでるなんてことはありませんよね）。でもってコマンドモードで，

A＞HOSI

と打ってしばらく待つと……。X68000のディスプレイは宇宙船の窓となってあなたは星の世界を飛んで行く……，という次第でございます。んでもって何かのキーを押せばおしまい。

え，なになに，投稿原稿によれば，

「……これは，BASICでも簡単に組めるのですが，Cコンパイラを買ったからわざわざCで作りました。……利用法はまず，ただ見ているだけでよいのですが，ゲームのエンディングのバックに使ったり，特殊効果で仮にワープのときにトンネルの役割をしたりといろいろと役に立つと思います。あと，『r ＋＝10+rnd（）＊10』の前の10を変えることによって，いろいろなかたちに変わりますので暇な人は変えてみてください……」

だそうです。うーむ，えらいぞ，その見上げた根性。コンパイラを買ってしまったからには，とにかく使ってしまえ。うむうむ，こーでなくっちゃね。プログラム的にはちょっと短いし，なんかX-BASICっぽい感じ（コンパイラの場合の画面設定は間に合わせでも付けたほーがいいよ）ですが大変頑張ってくれました。拍手，拍手（パチパチパチ)。あ，ちなみにリストはちょっとばかしいじってしまいました。ごめんね。オリジナルの星の色のパレットコードは255だったので，もしそっちのほうがいいというのであれば直してあげてください（ほかにもちょっといじったけどね)。そうそう彼は変数表も付けてくれました。

変数表

X，Y	：星が出る位置を表す
X1，Y1	：星を描くときの（X，Y）の増加量を表す
R	：描くときの角度を表す
C	：カラーを表す
I	：半径を表す
D	：ループに使う

うむうむ，よい心掛けじゃ。皆も見習うように。

さて，夏とはいえ夜は涼しいし（昼に比べるとね)，星でも見上げながらプログラミングに精を出しますか。うっ，しまった。わし，近眼で星がよく見えない……。

おあとがよろしいようで……，つくてんてんてん。

リスト2　なさけない★星★

```
 1:     /*  ★ 星 ★  */
 2: #include <basic0.h>
 3: #include <conio.h>
 4: #include <stdlib.h>
 5: #include <math.h>
 6: #include <graph.h>
 7: #include <iocslib.h>
 8:
 9: void main(b_argc,b_argv)
10: int b_argc;
11: char *b_argv[];
12: {
13:   register int x,y;
14:   register int x1,y1;
15:   register int r=0,c,i,d;
16:
17: /*  初期設定  */
18:   screen(1,2,1,1);
19:   console('NASI','NASI',0);
20:   b_csw(0);
21:   cls();
22:   vpage(0);
23:   apage(0);
24:   printf("おまちください！");
25:
26: /*  画面を描く  */
27:   while( r<360 )
28:   {
29:     r += 10+rnd()*10;
30:     x= 256;  y= 256;  i= 0;
31:     x= x+cos(r*pi()/180)*4; /*追加部分  */
32:     y= y+sin(r*pi()/180)*4; /*  その1  */
33:
34:     while( 1 )
35:     {
36:       c= ((c+1) % 30)+1;
37:       i += 1;
38:       x1= cos( r*pi() / 180 ) * i;
39:       y1= sin( r*pi() / 180 ) * i;
40:       line( x , y , x+x1 , y+y1 , c , 0xffff );
41:       x += x1;        y += y1;
42:       if ( x<0 || x>511 || y<0 || y>511 ) break;
43:     }
44:   }
45:
46:           /*  パレットで色をかえる  */
47:
48:   cls();
49:
50:   for( c=1 ; c<31 ;c++)
51:   {
52:     palet(c,0);    /*追加部2*/
53:   }
54:
55:   vpage(1);
56:   while( kbhit() == 0)
57:   {
58:     for( c=1 ; c<31 ; c++)
59:     {
60:       palet(c,0x841e);
61:       for( d=0 ; d < 2000 ; d++);
62:       palet(c,0);
63:     }
64:   }
65:   screen(2,0,1,1);     /*追加部3*/
66:   OS_CURON();
67: }
```

(で)のぱーてぃハンズ――――(その6)

ああ,感動の第1部　完

はいはい，皆さんこんばんは。ぱーてぃハンズの時間です。

さて，今月はぱーてぃハンズ第1部「ゼビモドキ編」の最終回としてゼビモドキゲームの当たり判定のプログラムの解説をやってしまうのです。ずーっとこのコーナーを読んでくれているニコちゃんの皆さんはご存じかとは思うのですが（ちなみに，ニコちゃんの対語としてコマッタちゃんがあるというのは私たちの世代にしかわからない，どーしょもない余談だったりする），当たり判定には（で）のショートプロぱーてぃ５月号に掲載のsp__chk ( )を使ってしまうのです。うーん，手抜き手抜き。ちなみにsp__chk ( )は６月号の Oh!X 特製ディスクにも入っていますので打ち込んでいない人はそちらから持ってきてくださいね（６月号もないという人は……，私は知らない……）。

で，当たり判定なのですが，知ってのとおり，もし，これがなかったらいつまでも点は入らん，自機は死なん，意味もなく永遠に空間をさまよわなくてはならんという，シューティングゲームにはなくてはならない，まるでコーヒーにクリープ（ふっ，古い），お好み焼きにオタフクソース，SX-WINDOWに2MバイトRAMのような存在なのです。

世の中のゲームの当たり判定というといろいろあるのですが，今回どのような方針でいくかというと，

「自機のタマに敵が当たっているかどうかのチェック」

「自機に敵or敵のタマが当たっているかどうかのチェック」

の２回に分けて行ってしまうのです。なんでこぎゃんややこしーことをするんじゃい！　という方もいらっしゃるかもしれませんが，はっきりいって，「sp__chk ( )がそーゆー仕様になっているから」というだけの理由なんです。他意はありません。

もう少し詳しく説明するとsp__chk ( )の持っている機能というのが，

「指定された１個のスプライトとほかのスプライトが重なっているかどうか」

を見るルーチンだから，敵についてと自機についてと１回１回分けて考えてやらなきゃいけないんですね。これがたとえばアーケードマシンみたいに，どれでもいーから当たったら教えてねというスプライトの衝突割り込みがあるマシン（あるいはソフト）だとそれ用の割り込みルーチンを作ってあとは果報は寝て待てという３年寝太郎方式も使えるのだけど。ま，これはこれでよしとしましょう。そーゆーわけです。

屋根まで飛んで,こわれて消えた

いま思ったんだけどシャボン玉の歌って，なんかとってもシュールじゃありません？　深い意味はないけど。

ま，それはいいとしてまずはタマの当たり判定ですね。タマの当たり判定とひとことでいってしまうとそれしかやらないような気もしますがそんなことはありません。シューティングゲームのへそだけあっていろいろやることがあるんです。まず，タマの当たり判定をします（あ，当たり前ですね）。1110行でsp__chk()を使ってますね。ここでループを組んでいるのは，このゲームではタマが１画面中に３発出せるんで，それを１発ずつチェックしにいってるんですね。で，当たっていた場合は，

dafire ( )

というサブルーチンに飛びます。ここで当たっていた場合の処理をするんですね。

ところで，シューティングゲームで敵にタマが当たったときにはどうなります？　うんうん，敵が消える。それからスコアが増える。え，パワーアップアイテムが出たり，敵が膨れちゃったり，挙げ句の果てが花火が打ち上がったりする？　うん，まあ，そういうのもありますけどね……。

まず最初に，自分のタマと敵が当たったわけですから，とりあえず自分の撃ったタマを消します。これはタマのｘ，ｙ座標に(０，０)を入れておけばタマの表示部分が勝手に消してくれますね(一応，念のためにいっておくけど，スプライトってちょっと座標が変わっていて，(０，０)におくとスプライトは消えてくれます。知ってますよね？)。で，当たったよ，というのがわかるように爆発パターンを表示して，敵の座標も(０，０)にします(1210，1220行)。こうすれば敵の表示もあとで消してくれるし，その間爆発も残ってくれるしで一石二鳥なわけです。とりあえず，これだけ。「スコアが出ないのはともかく，敵に当たったときの効果音もないのはとんでもない手抜きじゃないかっ！」といわれても「そーです，手抜きなんです」と答えるしかないようなぐらいに何もしてませんが，そこは皆さん，これを踏み台にして頑張って本ゼビなりギャラッパなりにしていただきたいのでした。

あっ,上がった。たーまーやー！

でもって，自機の当たり判定ルーチンatarim ( )です。こっちはさらに手抜きです。なんてたって自機のストック（自分の命は３つまでというインベーダーからの伝統のあれね）とかという制度もなく，当たったらいきなり死んでしまってゲームオーバーなんですから。爆発パターンさえありません。世の中なんてこんなものよ。うむうむ。やっていることは，自機が当たっていたらゲームオーバーの表示を出してBGを消して，そのままストップ。これだけです。例によって皆さんでどうにかしていただくのでした（しかし，１/４ページで終わりそうとかいっといて今月も結構書いたなあ……）。

……ああ，苦闘６カ月。ついにすべての作業が終わり，ゼビモドキが完成しました。ふーっ。しかしなんですね。終わったからこんなことがいえるのかもしれないけど，やっぱりもうちょっといろいろな機能をつけたかったですね。やっぱり自機は３機……いや，個人的な好みとしてはダメージ制にしたかったし，それにはやっぱりスコアも欲しかったし，あと，背景をスクロールさせたりとか，BGかグラフィックに絵でも描いてボスキャラを作ったりとか（動かすのはスクロールでできるから簡単だしね），音楽は無理としてもせめて効果音ぐらいは入れたかったなあ。うむむむ。まあ，そのうち気が向いたら，またゼビモドキ２にバージョンアップしたい……。

ということで，これにて，

第１部　未完

なのであった。それではまた来月Oh!Xで。あちょっ！（と瓦を割って去る）

```
 190 atarim()
 410    atarif()
 450    atarif()
 920 func atarim():/*自機当たり判定*/
 930  if sp_chk(33,38,51)<>0 then{
1010                  locate 12,8
1020                  print"gameover"
1030                  bg_set(0,0,0)
1040                  stop
1050 }
1080 endfunc
1090 func atarif():/*タマ当たり判定*/
1100 for i=0 to 2
1110    j=sp_chk(34+i,38,40)
1120     if j<>0 then  dafire()
1160 next
1170 endfunc
1180 func dafire()
1190 fire_x(i)=0:fire_y(i)=0
1200   sp_set(34+i,fire_x(i),fire_y(i),&H120)
1210   sp_set(j,enemy_x(j-38),enemy_y(j-38),&H124)
1220   enemy_x(j-38)=0:enemy_y(j-38)=0
1230   endfunc
```

ハードウェア工作入門《3》

基本インタフェイス回路 その3

Misawa Kazuhiko
三沢 和彦

インタフェイス回路の製作はうまくできましたか。今回はソフト編で，作った回路を制御するプログラムを書いてみましょう。まず基本となるドライバを作り，その上で走る応用プログラムを書くというのはどんな機器を使う場合でも同じです。

今回は自作I/O基板をX68000から制御する方法を考えましょう。まず代表的なパラレルインタフェイスであるジョイスティックポートを制御するための一般的な説明から始まって，実際にX-BASICでジョイスティックポートにつないだI/O基板をコントロールするための外部関数をアセンブラで記述してみます。そして，その外部関数を使った応用プログラムを例にとりながら，X68000で外部機器を制御することへの入門となるよう解説していこうと思います。

データのやりとり

外部I/O機器は複数の装置がつなげるように，各々にアドレスを設定して区別します。ある特定の装置にアクセスするにはそのI/Oアドレスを指定してデータを読み書きします。X68000の場合はメインメモリとI/Oのアドレス空間は共通なので，マシン語で記述するときはI/Oアクセスでも普通のメモリアクセスと同じです。やりとりするデータは８ビットの論理データです。すなわち，外部I/Oにつながっている各ビットのデータ線がそれぞれHかLかをとり，それらを８本まとめてCPUが処理するというわけです。

ただしX68000の場合は，メモリ空間全体がユーザー領域とスーパーバイザ領域のどちらかに分かれていて，通常はスーパーバイザ領域にはアクセスできないようになっています。これは，OSなどをスーパーバイザ領域に置いて，プログラムのバグなどによってコンピュータのシステム全体を破壊しないようにするためです。I/O空間にもスーパーバイザ領域に割り当てられているものがあり，そこにアクセスするには工夫が必要です。今回のメインとなっているジョイスティックポートもスーパーバイザ領域にあります。

このあたりの予備知識を頭に入れておいたうえで，外部関数のソースリスト（リスト１）を一緒に見ていきましょう。

I/O制御用外部関数

外部関数のアセンブラソースには大きく分けてヘッダ部と定義ルーチン本体とがあります。ヘッダ部は今回の本題から外れるので，簡単な解説をコラム１に譲ることにし，さっそくドライバ本体部分の説明に移ります。

外部関数はポートからのデータ入力がioinp関数，出力がioout関数として定義されています。ioinpは引数なしで，戻り値は入力データになっていて，iooutは引数に出力データをとり，戻り値はありません。

ioinp関数が行っているのは，ジョイスティックポートのアドレスを指定して，そこから８ビットデータを読み込むだけのことです。ソースに沿って見ていきますと，まずジョイスティックポートの入力のアドレス（$E9A001）をportaというシンボルに割り当てています。次のioinp_execからメインルーチンがスタートですが，ここでいきなりサブルーチン_ioinpに飛んでいます。これは，X-BASICの外部関数とBASTOCのライブラリとを１つのソースファイルで兼用させるための工夫なのですが，詳しくはコラム２を参照してください。

さて，メインルーチンの_ioinpに入ると，まず先ほど説明したように，スーパーバイザ領域にあるジョイスティックポートにアクセスするためにDOSファンクションコールの_SUPERによって，スーパーバイザ領域にもアクセスできるモードに入ります。

そして，あらかじめクリアしておいたD１レジスタにI/O基板から入力してきたportaのデータを転送します。目的が達成されたら再び，元のユーザーモードに戻してから，D1レジスタ内のデータを外部関数からの戻り値を格納するバッファに入れます。

コラム１　外部関数ヘッダ部の解説

X-BASICの外部関数の作り方は1987年8月号に柔野氏による詳しい解説がありますが，ここでもう一度ソースリストに沿って復習してみたいと思います。

最初の6行はアセンブラソースを書くときのおまじないのようなものです。別のアセンブラソースを挿入するためのinclude文，BASTOCのライブラリとして使うサブルーチン名を指定するglobl文，ソースをテキストベースに指定するtext文，そして実際に機械語コードに置き換わる領域の始めにメモリ境界を偶数にするeven文からなっています。

次に外部関数のヘッダ部を順番に説明していきましょう。初めに，インフォメーションテーブルですが，ここには，11個のアドレスが並んでいます。最初から8個は特別な処理を行う関数のために設定するルーチンのアドレスを指しているものですが，通常は使用しないため，rtsのみのダミールーチンになっています。

9番目は関数の名前を定義したテーブルの先頭アドレス（ptr_token）を指しています。今回のI/Oドライバは，データ入力をioinp，出力をiooutにしています。ptr_tokenで始まる関数名テーブルにこの２つが並んでいるのがわかるでしょう。

10番目は関数名テーブルで定義したそれぞれの関数の引数および戻り値の型と順番とを定義するテーブルの先頭アドレスを指しています。すなわち，このアドレス（ptr_param）の先を見ると，ioinp_parとioout_parの２つのアドレスが並んでいて，それぞれが以後のパラメータIDテーブルのアドレスを指していることになります。そして，パラメータIDテーブルにそれぞれの関数の引数と戻り値が実際に定義されているわけです。これより，ioinpは引数がなくて，ポートから入力したデータに当たる整数型の戻り値を持ち，iooutはポートへの出力データに当たる整数の引数が1個で戻り値はないことがわかります。

11番目はそれぞれの関数の実行ルーチンのアドレスを並べたテーブルの先頭アドレスを指しています。このptr_execの先にあるioinp_execとioout_execとが実行ルーチンの先頭です。

以上でヘッダ部は終わりです。駆け足での復習でしたが，自分で外部関数を定義する必要がなければ，この部分はあまり理解しなくてもよいと思います。

戻り値バッファはプログラムの最後にretdatで定義された10バイトの領域になっています。今回の関数では戻り値は整数型なので，ポートから入力したデータは最後の4バイトに当てはめます。実際有効なのは最下位1バイトなのですが，一応ここでは4バイト整数として扱っています。

ところで，外部関数の戻り値の受け渡し方ですが，A0レジスタに戻り値バッファの先頭アドレスを入れてrtsでリターンしてやるだけでOKです。このプログラムではいったん_ioinpルーチンからioinp_execルーチンに戻ってきてから，改めて外部関数処理から出るrts命令がきています。なお，外部関数を正常に終わるときはD0レジスタに0を入れてからリターンする必要があります。

iooutt関数については，引数があるので注意が必要です。引数である4バイト整数(実際ここでも，有効なのは最下位1バイトのみ）をまずジョイスティックポートの出力のアドレス($E9A001)をportcというシンボルに割り当てています。整数型の引数はスタックの先頭から13バイト目から積まれているので，そこから読み出してポートに転送するわけですが，ここではX-BASICの外部関数とBASTOCのライブラリとを1つのソースファイルで兼用させるためにいったん引数をスタック上で積み替えて，_iooutルーチンに飛んでいます。

_iooutルーチンに入ったら，まずスーパーバイザモードに切り替え，スタックにある出力データをD2レジスタに移してからそのデータをポートに出力します。そして，再びユーザーモードに切り替えて終わります。ioout関数は戻り値がないので，終わり方は簡単です。

それでは，アセンブラのソースをエディタで入力したのち，iodrv.sのファイル名でセーブしてください。あとは，

```
cc iodrv.s
```

でiodrv.oとiodrv.xを生成し，このiodrv.xをBASIC2のディレクトリの中にiodrv.fncのファイル名でコピーします。もちろんBASIC.CNFファイルの中に

```
FUNC = IODRV
```

と加えることも忘れずに。

サンプルプログラム

これで，外部関数がX-BASICのなかで使えるようになりました。まず簡単なサンプルをリスト2に挙げます。これは16進ロータリースイッチの入力をそのまま7セグメントLEDに出力するプログラムです。もちろんX68000自体がデータの読み書きをしているわけですから，データはBASIC上の変数に取り込まれていて，それがそのまま画面上に表示されます。なお，7セグメントLEDの出力は3ビットなので，0～7しか表示されません。

このプログラムの中にinpvalとoutvalの2つの定義関数がありますが，これについて少し説明します。そもそも，X68000とI/O基板とのデータのやりとりは，数値データではなくて論理データです。

たとえば入力データは，ロータリースイッチから入力した数値（0～15）を2進数で表し，下の桁から1が立つビットがL，0が立つビットがHに対応しています。いま入力データが9とすると2進数で1001_Bですから，入力ポートから読み込む論理データはLHHLとなります。通常はHのビットが1，Lのビットが0に形式的に対応しますので，ioinp関数で読み込む値は6（$=0110_B$）ということになります。inpval関数はこのように各ビットの論理が反転しているデータを正しい値に直すための関数です。

次に出力データはパラレルインタフェイスの8ビットデータのうち，ビットNo.4，6，7の3ビットを使っているのですが，

これを今回のI/O基板では3ビットの2進数データとして，上の桁からビットNo.4，7，6の順に変則的に並べて使っています。しかもビットNo.4はデータとして1を出力するとポートにH，0を出力するとLがきちんと出ますが，ビットNo.6と7は1を出力するとL，0を出力するとHが出て，ビットNo.4と出力が反転しているのです。したがって，たとえば5を出力したいときは，101_Bですから下2桁は反転してHHLとし，しかもビットを入れ替えて，HLXHXXXX（Xは関係ないビット）の論理データを出力しなければなりません。

outval関数は引数として0～7を取り，

コラム2　外部関数とライブラリの兼用

X-BASICの外部関数とBASTOCのライブラリとを1つのソースファイルで兼用させるための工夫について説明する前に，コンパイルしたときに外部関数がどう扱われるか説明しておきましょう。

外部関数の処理の部分は外部関数名の前にそのまま“_”をつけたラベルのサブルーチンを呼び出します。したがって，コンパイルされたマシン語プログラムはそのラベルのサブルーチンをライブラリとして持つ必要があります。

たとえば今回の場合，BASICプログラム中でioinp関数を呼び出す箇所は，

```
bsr _ioinp
```

という命令に書き換わります。そして，引数は外部関数におけるカッコの中の順番のとおりにスタックに積まれます。それに対して戻り値はD0レジスタに格納されてサブルーチンから戻ってきます。

これだけの予備知識を頭に入れて，もう一度ソースを見てください。ioinp関数とioout関数の定義ルーチンの中にそれぞれの関数名に_をつけた_ioinpと_iooutというサブルーチンがあるのがわかるでしょう。このソースをライブラリとして使った場合，このサブルーチンが参照されるのです。ですからこのサブルーチンにおける引数，戻り値の処理は上で述べたとおりになっています。

たとえば，_ioinpの戻り値については，_ioinpルーチンの最後で，入力ポートから読み込んできてD1レジスタに格納したデータをD0レジスタに移してからリターンしています。また，_iooutの引数もスタックのいちばん上に積んである（実際は，引数をスタックのいちばん上に積んだあと，サブルーチンコールをして，そのリターンアドレスがさらにスタックに積まれるので，上から2つ目の整数データが引数になっている）データをD2レジスタに取り出して，出力しています。

さて，X-BASICの外部関数の引数と戻り値に関してはライブラリのサブルーチンコールと形式が異なるので，外部関数を呼び出すときは，その部分を変換してやるのです。具体的にiooutt関数の引数については，スタックの13バイト目に積まれているデータをもう一度スタックのいちばん上に積み直してライブラリのサブルーチンコールと見かけ上同じ形式にしてから，共通の_iooutルーチンを呼び出すのです。

このとき，最後に外部関数から出るときのリターンアドレスが積み直したデータによってスタックの中に埋もれてしまい，このままでは正常に終了しないので，スタックポインタを操作してから終了しています。また，ioinp関数の戻り値については，戻り値バッファに格納してやればよいので特別な操作は必要ありませんが，外部関数を正常に終わるときはD0レジスタは0でなくてはならないので，_ioinpルーチン内で戻り値を格納したD0レジスタを改めて消去しておかなければなりません。

7セグメントLEDが正しい値を表示するようにデータを変換したあとにポートに出力するものです。

皆さんがもし自分で応用プログラムを組むときは，このinpval関数，outval関数をそのまま引用すると便利でしょう。なお，このプログラムをコンパイルするときは，

CC IOTEST.BAS IODRV.O

を実行してください。以後，すべてのプログラムについて，コンパイルのときはIODRV.OをリンクすればOKです。

応用プログラム

先に述べた，inpval関数とoutval関数を使ってさらにゲームを組んでみました（リスト3)。内容は簡単そのもの，7セグメントLEDの数字が回るルーレットです。ロータリースイッチでスタート，ストップを行いますが，それだけではあまりにも色気がないので，ルーレットの数字当ても加えてみました。このあたりのゲームデザインは皆さんのほうが得意かもしれません。

わざわざ自作のI/O基板を使わなくてもX68000単体でこの程度のプログラムは組めるわけですが，やはり，自作の基板をパソコン本体からコントロールする面白さもまた格別です。このような基本的な回路についてしっかり押さえておくことが，より高度な回路を使いこなしていくためには大切なのです。次回はもっと歯応えのある(しかも実用的な) 回路を扱いますので，楽しみにしていてください。

リスト1　I/O制御用外部関数iodrv.s

```
  1: ****************************************************
  2: *                                                  *
  3: *       ハードウェア工作入門  I/Oドライバ         *
  4: *          外部関数  ioinp & ioout                 *
  5: *                                                  *
  6: *      Ver. 1.0    1990. 6.17  K. Misawa           *
  7: *                                                  *
  8: *        ioinp()     : データ入力                  *
  9: *                      (引数)   なし               *
 10: *                      (戻り値)データ              *
 11: *        ioout(data) : データ出力                  *
 12: *                      (引数)   データ             *
 13: *                      (戻り値)なし                *
 14: *                                                  *
 15: *       X-BASIC外部関数                            *
 16: *        &  BASTOCライブラリ  兼用                 *
 17: *                                                  *
 18: ****************************************************
 19:
 20:                 .include        doscall.mac
 21:                 .include        fdef.h
 22:                 .globl          _ioinp
 23:                 .globl          _ioout
 24:                 .text
 25:                 .even
 26:
 27: ****************************************************
 28: *                                                  *
 29: *                 外部関数ヘッダ部                 *
 30: *                                                  *
 31: ****************************************************
 32:
 33: *
 34: *インフォメーションテーブル
 35: *
 36:                 dc.l            x_init
 37:                 dc.l            x_run
 38:                 dc.l            x_end
 39:                 dc.l            x_sys
 40:                 dc.l            x_brk
 41:                 dc.l            x_ctrl_d
 42:                 dc.l            x_res1
 43:                 dc.l            x_res2
 44:                 dc.l            ptr_token
 45:                 dc.l            ptr_param
 46:                 dc.l            ptr_exec
 47:                 dc.l            0,0,0,0,0
 48:
 49: x_init:
 50: x_run:
 51: x_end:
 52: x_sys:
 53: x_brk:
 54: x_ctrl_d:
 55: x_res1:
 56: x_res2:
 57:
 58:                 rts
 59: *
 60: *関数名テーブル
 61: *
 62: ptr_token:      dc.b    'ioinp',0
 63:                 dc.b    'ioout',0
 64:                 dc.b    0
 65:
 66:                 .even
 67:
 68: *
 69: *パラメータテーブル
 70: *
 71: ptr_param:      dc.l    ioinp_par
 72:                 dc.l    ioout_par
 73:
 74: *
 75: *パラメータIDテーブル
 76: *
 77: ioinp_par:      dc.w    int_ret
 78: ioout_par:      dc.w    int_val
 79:                 dc.w    void_ret
 80:
 81: *
 82: *実行アドレステーブル
 83: *
 84: ptr_exec:       dc.l    ioinp_exec
 85:                 dc.l    ioout_exec
 86:
 87:                 .even
 88:
 89: ****************************************************
 90: *                                                  *
 91: *                 定義関数ルーチン                 *
 92: *                                                  *
 93: ****************************************************

 94:
 95: *
 96: *データ入力関数  ioinp()
 97: *
 98: porta           equ     $e9a001
 99:
100: ***実行アドレス
101: ioinp_exec:     bsr     _ioinp
102:                 move.l  #0,d0
103:                 rts
104:
105: ***メインルーチン
106: _ioinp:
107:
108: ***スーパバイザモードに入る
109:                 clr.l   -(sp)
110:                 dc.w    _SUPER
111:                 addq.l  #4,sp
112:                 move.l  d0,spbuf
113:
114: rd_ok:          clr.l   d1
115:
116: ***ジョイスティックポートから読みだし
117:                 move.l  #porta,d2
118:                 movea.l d2,a3
119:                 move.b  (a3),d1
120:
121: ***ユーザーモードに戻る
122:                 move.l  spbuf,-(sp)
123:                 dc.w    _SUPER
124:                 addq.l  #4,sp
125:
126: ***戻り値をバッファに格納
127: rd_ready:       move.l  d1,int_data
128:                 lea.l   retdat,a0
129:                 move.l  d1,d0
130:
131:                 rts
132:
133: *
134: *データ出力関数  ioout(data)
135: *
136: portc           equ     $e9a005
137:
138: ***実行アドレス
139: ioout_exec:
140:
141: ***引数をスタックに積み替え
```

```
142:                 move.l  12(sp),d1
143:                 move.l  d1,-(sp)
144:
145:                 bsr     _ioout
146:                 addq.l  #4,sp
147:
148:                 move.l  #0,d0
149:                 rts
150:
151: ***メインルーチン
152: _ioout:
153:
154: ***スーパバイザモードに入る
155:                 clr.l   -(sp)
156:                 dc.w    _SUPER
157:                 addq.l  #4,sp
158:                 move.l  d0,spbuf
159:
160: ***ジョイスティックポートへ書き込み
161: wr_ok:          move.l  #portc,d2
162:                 movea.l d2,a3
163:                 move.l  4(sp),d1
164:                 move.b  d1,(a3)
165:
166: ***ユーザーモードに戻る
167:                 move.l  spbuf,-(sp)
168:                 dc.w    _SUPER
169:                 addq.l  #4,sp
170:
171:                 rts
172:
173: *
174: *スタックバッファ
175: *
176: spbuf           ds.l    1
177:
178: *
179: *戻り値格納バッファ
180: *
181: retdat:         dc.w    0
182:                 dc.l    0
183: int_data:       dc.l    0
184:                 end
```

リスト2　サンプルプログラムiotest.bas

```
 10 /* save "¥basic¥iotest.bas
 20 /*
 30 /*I／O基板用サンプルプログラム
 40 /*
 50 /*    1990.7.1  K. Misawa
 60 /*
 70 int v
 80 while 1
 90   v=inpval()
100   outval(v)
110   print v
120 endwhile
130 end
140 /*
150 /*データ入力
160 /*(引数)なし
170 /*(戻り値)ロータリスイッチの値
180 /*          0－15
190 /*
200 func int inpval()
210   int v
220   v=&B1111-(ioinp() and &B1111)
230   return(v)
240 endfunc
250 /*
260 /*データ出力
270 /*(引数)整数値
280 /*(戻り値)なし
290 /*(機能)引数を8で割った余りを
300 /*        LEDに表示
310 /*
320 func outval(d0;int)
330   int v,v0,v1,v2
340   v0=1-(d0 and 1)
350   v1=1-(d0 and &B10)/&B10
360   v2=(d0 and &B100)/&B100
370   v=&B10000000*v1+&B1000000*v0+&B10000*v2
380   ioout(v)
390 endfunc
```

リスト3　応用プログラムroulette.bas

```
 10 /* save "¥basic¥roulette.bas
 20 /* save@"¥basic¥roulette.doc
 30 /*
 40 /*I／O基板用応用プログラム
 50 /*               ルーレットゲーム
 60 /*
 70 /*    1990.7.4  K. Misawa
 80 /*
 90 width 64
100 /*
110 m_init()
120 m_alloc(1,100)
130 m_assign(1,1)
140 m_tempo(120)
150 /*
160 int i=0,j
170 /*
180 /*初期チェック
190 /*
200 while inpval()<>0
210   locate 10,14
220   print "ロータリースイッチを0にして下さい!"
230   for iii=1 to 2000 : next
240   cls
250 endwhile
260 /*
270 /*予想の入力
280 /*
290 cls
300 repeat
310   locate 10,14
320   input "予想する数を入力して下さい";j
330 until j>=0 and j<=7
340 /*
350 /*ルーレット
360 /*(ロータリースイッチを動かすと停止)
370 /*
380 cls
390 locate 10,14
400 print "ロータリースイッチを回すとルーレットは止まります。"
410 repeat
420   i=(i+1 and 7)
430   outval(i)
440   for iii=0 to 200 : next
450 until inpval()<>0
460 /*
470 /*結果判定
480 /*
490 cls
500 locate 10,14
510 if i=j then { win() } else { lose() }
520 m_play()
530 end
540 /*
550 /*データ入力
560 /*
570 func int inpval()
580   int v
590   v=&B1111-(ioinp() and &B1111)
600   return(v)
610 endfunc
620 /*
630 /*データ出力
640 /*
650 func outval(d0;int)
660   int v,v0,v1,v2
670   v0=1-(d0 and 1)
680   v1=1-(d0 and &B10)/&B10
690   v2=(d0 and &B100)/&B100
700   v=&B10000000*v1+&B1000000*v0+&B10000*v2
710   ioout(v)
720 endfunc
730 /*
740 /*結果の処理
750 /*
760 func win()
770   print "大当たり!"
780   m_trk(1,"@57v15o5 f8e8d8c8f8e8d8c8f4a4g4r4")
790 endfunc
800 /*
810 func lose()
820   print "残念でした"
830   m_trk(1,"@34v15o3 f4c8c8d4c4r4e4f4r4")
840 endfunc
```

超能力実験の成果

Mr.マリックに埋もれた超能力

新宗教，オカルト，超能力，占いなどが，世紀末のいまにふさわしく，流行っているようです。神秘的なものが，人々を，特に若者を熱中させるという社会現象，文化現象に注目し，それを解析することはきわめて面白いことだと思います。最近読んだ別冊宝島（文献1）はそのような現象を取り扱った本です。新聞，テレビなどを賑わせている宗教などを豊富でかつ具体的にレポートしており，しかも，いろいろな方面からの分析はあっという間に読み終えてしまうほど興味をひくものです。

新宗教，オカルト，占いなどはあまりにもおどろおどろしいので，ここでは超能力を取り上げます。知能機械概論というタイトルにも，まあふさわしいといえるでしょう。とにかく気をつけることは，真面目にかつ知的に，つまり科学的な態度を持ち続けることでしょう。

人工知能，あるいは計算機と超能力にいったいどんな関係があるんだ，と真面目に質問されるとちょっと苦しいこともありますが，まずこう答えるでしょう。「人工知能は人工的に作り出された人間の知能を呼ぶものだ。その人間の知能に含まれる（かもしれない）超能力も当然科学的に見極めて対象とすべきだ」。

また，超能力をはじめとする神秘的なものに向かう若者の心と，計算機に向かう若者の心に間接的な関係ではあるが，案外大きな共通点があるのではないか，という印象を僕が持っていることも両者を関連づけようとするひとつの動機になっているかもしれません。

でも実際のところ，なぜ超能力のことを取り上げたかというと，最近，超能力について学問的に取り上げた（そして科学史の分野ではきわめて有名な村上陽一郎氏の訳した）1冊の野心的な本を読んだということ（これがきわめてセンセーショナルっぽいのだが），それに一見超能力っぽいという現象を目撃したからなのです。

科学思想家の書いた1冊の本

超能力を扱った本というと，スプーン曲げとか，Mr.マリックとかいうような類の事象や人を扱った，半分お遊びのような本を連想してしまいます。しかし，そうではないまれな1冊の本（文献2）がなぜか自分の家の本棚に存在していました。

著者のケストラーはもともとは新聞記者でしたが，政治活動を行い，刑務所に入れられた後，科学思想に関する著作を発表し始めました。有名なものには，「ホロン革命」「機械の中の幽霊」「還元主義を越えて」などがあります。ホロンに基づく階層構造などの話はかなり日本でも話題を集めました。

彼の科学思想の特徴は次の点にあるようです（文献3）。

1）現実生活から科学を遊離させず，抽象的な理論に走らない。

2）正統的な思想に反逆する（反ダーウィニズム，超心理学への注目）。

3）専門家では言いにくいことでもズバズバ言う。

本書はただ単に超能力が存在すると主張しているのではありません。超能力を現代物理学，心理学，生物学などの最先端の成果と関連づけようとしているのです。ただし，ケストラー自身は科学者ではないので，自分でなにか実験を行ったり，論証したりするのではなく，第三者的なジャーナリストに近いスタンスをとっています。

本書では，まず，テレパシー，予知，念動などのような現象は確実に存在するものであるということを，さまざまな研究者たちの発表を具体的に提示することで，我々を説得しようと試みます。そして，次には，がらりと話題が違うように思われる，現代量子力学における，人間の想像を越える奇妙な素粒子たちの姿について述べます。読者はしだいに，もしかしたら最初に述べた超能力をこのような物質のミクロな姿で説明をしようとしているのだろうかと感じ始めるでしょう。

次にケストラーは超能力を完全に説明するまでには至らないものの，いくつかの手掛かりを提示します。それは現在の学問体系の大きな基盤である「因果関係」の絶対的存在を否定するものとしての「連続性」「同期性（シンクロニシティ）」の概念です。これらは生物学，物理学，心理学のそれぞれの巨匠であるカメラー，パウリ，ユングらによって考え出されている概念です。

そして，本書ははっきりした結論を提示するには至らないまでも多くの示唆を残して終わります。ここでは，本書全体の主張までは踏み込まず，超能力の存在に関するデータを突きつける第1章だけについてもう少し紹介することにします。

信じざるを得なくなる第1章

超能力の学問的研究の先駆者はアメリカのデューク大学心理学科のライン助教授だそうです。彼は1932年に超心理学研究所を設立し，それまでは怪しげなところの多かったこの手の現象に対して厳密な科学的手法を導入して解析しようとしました。

トランプ当てやサイコロ振りの実験はなんと何千人という無作為に抽出された人たちを使い，何百万回という実験を繰り返したものだそうです。彼の示した超能力の存在を証拠づける結果は，いまでは議論の余地のない完全なものであるとされているとのことです（ケストラーの解釈だといわれればそのとおりかもしれませんが）。

彼の実験も含めて数多くの結果が第1章には記されていますが，ラインの行ったテレパシーに関するひとつの実験を紹介することにしましょう。「ゼナーカード」と呼ばれる，円，正方形，十字，星，波というわずか5種類のカードを用います。その1枚を送り手が無作為に選び，それを見て受け手にイメージを送り，受け手はそれが何であるかということを当てるのです。当然，テレパシーなど存在しないのならば，それが当たる確率は5分の1，つまり100回行えばだいたい20回当たるはずです。

1934年にラインによって公開された記録は，特定の（超能力を有していると考えられる）被験者に関する，8万5千回にも及ぶ実験のすべてのデータでした。これによると，全体を通じて，100回に対して 28回という割合で当たり続けたということです。これは偶然と考えるにしては天文学的な数字のずれということです。

さらにいろいろな人の行った実験の結果が記されています。それらはイギリス心霊研究学会の学会誌に論文として発表されています。この学会は最初はなんとなく胡散臭く思いましたが，読んでいくうちにきわめて権威があるのではないかと感じられてきました。というのも，数ページをさいてこの学会の信頼性を高めるような記述があったからです。

この学会の歴代の会長（全部で47人）の中には３人のノーベル賞受賞者，１人の首相，10人の王立学会のメンバー，多数の大学教授（ケンブリッジ，ハーバード，オックスフォードなど）を含んでおり，本書にはわざわざ全員のリストも載せられています。さらに副学会長や運営委員なども含めると，ずいぶん権威のあるものになるであろうと述べています（たとえば電子の発見者のトムソンまで含まれているとのことです)。

日本の現在のこの手の研究がほんの少ししか行われていないとの認識を強めたのは次の記述を読んだときです。

「いまでは世界中どこを探しても，超心理学の研究学科を有する大学のまったくないような国を探すことがむしろ難しい」

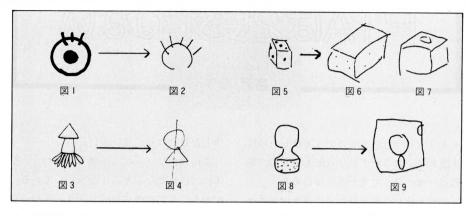
最近ではさらに厳密を極めた実験が行われているとのことです。それは，１個の電子がある放射線を発生する量子過程を利用して，４つのランプのうちのひとつを無作為点灯させ，被験者はどれがつくかを当てるというものです。２万回行って，偶然ならば10のマイナス10乗の確率というきわめて起こりにくい現象を確認したということです。

超能力実験の行われた日

さてやっと僕の身近で確認できたと思われる超能力についてお話しするときがきました。実はこの現象は僕が先ほどの本を読む前に偶然起こったわけで，厳密に実験という性格，あるいは客観的な説得力を持たせたものではないということは最初に言っておかなければなりません。しかし，記録として残すだけの価値はあると僕は思ったのです。

テーマはテレパシーです。方法はひとりが適当に簡単な絵を描きそれを見ながらイメージを送り，もうひとりはそのイメージを頭の中で受け取りそれを絵に描くというものです。

そのときの状況を忠実に追っていくことにしましょう。僕自身は同じ部屋で別の仕事をしており，途中で騒ぎが大きくなったので見物に加わりました。ですから，加わるまでのことや被験者たちの心情などについてはあとで聞いたことからまとめました。

------------超能力実験中継開始------------

1)　暇を持て余していたＹ，Ｔ，Ｓの３人の話題は偶然超能力に移り，それでは試しにテレパシーの実験でもしようかということになる。

2)　Ｙはどっかで見たようなマーク(図１)を描いて送る。Ｔはパッとイメージが湧きそれを描く（図２)。あまりの類似に驚き，もう１回やろうということになる。

3)　Ｙはイカの絵（図３）を描き，Ｔに送る。Ｔは四角形と三角形の絵が頭に浮かび，おでんだろうと（余計に）考えて，それを描く(図４)。結果に驚く。そのまま三角と四角を描けばよかったという話になる。話は盛り上がり次は立方体だろうなどという話題が間接的に出る（これは次の実験の正確さを落とすことになる，ただしサイコロという言葉はもちろん出ていない)。

4)　(興味が出て加わった)ＳはＴとＹにサイコロの絵（図５）を送る。豆腐だとＹが言い出し(図６)，それを聞いたＴは一部が尖った豆腐のイメージを受け取る（図７)。室内はすでに騒然とした雰囲気に包まれており，３人ともかなり冷静さを失っている。

5)　次にＹはまったく抽象的なイメージ（図８）を送るがＴは現実的なものをイメージしようとして，フロッピーディスクの絵（図９）を受け取ってしまう。当たらなかったということになるが，Ｙは（ちょっと強引だが）フロッピーの中心の円とその下の窓の部分そのものだと，解釈しようとする。

6)　その後，２回実験を行うがあまり似ていない（その後実物の絵は捨ててしまう)。もうこれ以上やっても，せっかくのこの驚きを打ち消すだけの結果になりそうな気がしてやめる。

------------超能力実験中継終了------------

この実験からすぐに結論を出そうとしているのでは，もちろんありません。読んだ方がひとつのデータとしてでも受け取ってもらえばいいのでしょう。実験結果は決められた記号や数字などを送るときのように，当たりはずれの成績が正確な数字で表されるようなものではありませんから，「こんなことがもし偶然起きるとしたらそれは，こんなに小さな数なんだ」ということもできません。しかし，自由に描ける絵だからこそ，これほど一致するのは驚異的なのだといえるような気もします。

４回の実験について，一応それぞれの成果を主観的に100点満点で評価してみましょうか。まず１回目はなんといっても95点。２回目は75点。３回目はパターンとしては90点だが，場の雰囲気が情報をかなり伝えてしまったのでマイナス50で40点。そして４回目はまあ20点でしょう。

お茶目な脱線

たまたま超能力の香りのする現象が身近に起き，そして，たまたま家の本棚にあった超能力に関する野心的かつ学問的な本を読みました。このような機会に恵まれたものとして，しかも科学者のひとりとして，今度は自分の方法で能動的に（ちょっとだけ）踏み込んでみたいと思います。熱い夏にはぴったりでしょう。ライン助教授の用いたものと同種の「ゼナーカード」もたまたま手に入ったことですし。

参考文献

1) 別冊宝島114「いまどきの神サマ」，JICC出版局，1990

2) 「偶然の本質」　アーサー=ケストラー（村上陽一郎訳)，蒼樹書房，1982(第５刷)

3) 現代思想６：特集　ケストラー現代科学への挑発，青土社，1983

第51回

猫とコンピュータ

PTAは2度死ねない

Takazawa Kyoko
高沢 恭子

小さい頃，いつも「いったいなんのためにあるんだろう」と不思議に思っていたPTA。それは子供だけでなく実際に参加しているおかあさんたちも感じているらしい。さあ，いまここにPTAの実態が明かされる！

もう少し冷たい夜明けが来てほしいのに，小鳥が飛び去るかすかな気配とともに，窓辺は一面，湿った光を浴びはじめる。

午前6時，毛皮を着た気の毒なホンニャアが，ベッドにしていた食卓の椅子から，コトンと降りる音がする。なぜかクーラーのきいた部屋を嫌って，リビングの気に入った場所で眠ることにしているのだ。

時計の音で目ざめるのかどうかわからないが，ホンニャアは6時になると寝室のドアの前にきて，「もう，起きてよネ」と声をかける。

さそわれてドアをあけ一歩踏みだす床からは，なまあたたかい湿気が上ってきて，今日1日の意欲と計画を，足元からくじけさせようとする。やっと梅雨の明けた7月なかば，ふと，これからやってくる盛夏を思うと弱気になる。あのひんやりした，さわやかな，はげましに満ちた朝はもうこないのかな。

でもだいじょうぶ，予期する猛暑はおそろしいけれど，現実に出会う真夏は楽しいものだ。いつも，新しい夏が待っている。

序盤戦

始まるまでが大さわぎ。これから何がおこるのか，どんなひどい目にあうことになるのかを，想像しているときがいちばんコワい。

春，PTAの各委員を，クラスの中から決める日はみんなコワくてつらい。つらいからこの日は出てこない人も多い。なぜつらいかというと，委員になるのをおたがいにゆずり合うのがつらい。

「ゆずり合う」とは，別の言い方をすると「逃れ合う」ことなので，逃れ合いのために，じぶんがどんなに忙しくて，委員としての時間を持つことが無理であるかを競争しなければならないのだ。

この競争はルールも基準もないから，根本的には無効なのをみんな承知している。だから，この競争に勝ったときもまたつらい。委員になってしまった人とじぶんと，どちらが時間的なゆとりが少ないかなんてとうていわかるはずもないし，時間のありなしで委員を「選出」するじぶんたちの失策について，けっして語れないのがつらいのだ。

「あの，校外補導委員というのをやります。それで，きょうはちょっと予定がありますので，帰らせていただきたいんですけど」

やむを得ない早退の交換条件として，いちばん無難に見える委員会の一員を志願した。とりあえずクラスからの「選出」を受けて，その委員会の構成員となり，委員長選を逃げきればいいのだ。その程度の任務なら，なんとかつとまるだろう。学年委員も，広報委員も，成人教育委員も，だいたいそんなぐあいにヒラ志望で各クラスから出陣していく。

半死半生

数日後，各委員会の初会合は正副委員長選出の巻。ここに来るのがやっとの思い，もうこの上はぜったい後に引かぬ，いや，前には出ない決意の面々が，それはそれはこわばった表情でにらみあう。

校外補導委員会は，委員長1名，副委員長5名を選ばなければならない。全校24のクラスから1名ずつ，24名の委員会だ。

インストラクター役の前年度委員長が，「まず各学年から2名ずつ，計6名を副委員長として選出，その中から1名を委員長としたらどうでしょう」と提案。

ただひとり，3年生のスイトウ（水藤）さんが「副委員長なら」と名乗りをあげただけで，どの学年も，長い長いうかがいあいがつづく。

「それではクジにしましょうか？」と前委員長。無策の切り札はいつもアミダくじ。スイトウさんの分をのぞいて，3年生は1名，ほかは2名ずつが泣き泣き決定。私はヒラ委員に逃れる。

そして6名の副委員長が委員長選出の話しあいを始め，ヒラの18名は申しわけなさとともに静観する。

ここで前委員長の善意のひとこと。

「3年生の親は，受験準備でなにかと多忙ですから，委員長からハズしてやったほうがよいと思います」

ご本人の体験からの発言だったが，結果的にはこれが波紋を呼んだ。委員長の選出範囲が1，2年生に限られ，該当者4人の戦慄が一気に高まったのは当たり前だ。

ただでさえ不信感でいっぱいの初顔あわせの会合に，不平等感が加えられて，委員長選は殺気をおびてきた。

長い重苦しい立往生のあと，とうとう1年生の副委員長から「3年生がやるべきでしょう！」と悲鳴に似た声があがった。

「なぜ，1，2年生の親が委員長にならなければいけないんですか。3年生がやって当たり前でしょう。私は副委員長を降りさせてもらいます」

そうなんだ，親が受験するのでもないのに，委員長をまぬがれるのはヘンだなと思っていた。でも，3年生がやるのが当たり前というのも少しちがう。

「私が委員長をやります」

3秒ほどあとにヒラのはずの私が挙手していた。中学校のPTAにはけっして関与すまいと，2年間逃げまわっていた私がやるべきだと，あっさり観念したのだ。

降りてしまった人は，「燃え尽き現象」

でいま社会に問題を投げかけている，多忙の花，看護婦さんだった。怒って当然だ。看護婦さんの非人間的重労働を，私たちと同列にしていたのが悪いのだ。

それにしても，そんなにみんなが嫌うPTAにしてしまったのは，いったい誰なんだ。幼稚園以来，愚かしい引き受けかた，押しつけられかたをしてきた，ニガぁい思いと共に，義務教育最終年の反省がおそってきた。

「お子さんヒトリ？ じゃできるじゃない」「おウチにいらっしゃるの？ だったらなされば」

誰もPTAの内容なんか考えていない。子供の数とヒマのぐあいで成り立ってるPTAが，いかにカラッポか，みんな知ってる。そうやって，じぶんたちでつまらなくしてしまったPTAに，みんなで意欲をなくしているのだ。

こうして，毎年，PTAは仮死状態でスタートする。

一部ハイテクで

始まってしまえば，コワいほどのことは何もなく，よくも悪くもレールが敷かれているのがPTAだ。このレールを切断させないことが，きっと最大の目標なのだろう。

行事計画のためにさっそく委員会を開こうと，名簿をたよりに電話をかけてみておどろいた。日中家にいる人なんて皆無だ。これではたしかに，委員の仕事はむずかしい。しかし，すべて夜を待っての進行というのも困ったものだ。

委員会の開催も土曜の午後の２時間がやっと，よほどスピーディに運ばないと行事はこなせそうにない。こんなことなら，みんなにラクをさせてあげようと，８割準備作戦に出る。

事前に計画案，実行案を文書でつくり，こんなものでいきましょうと，強引にすすめてしまうやりかただ。もともと前年度の行事を踏むのが基本だから，それほどの問題も起こらない。これでだいぶ時間の節約になる。エディタ，ワープロ，プリンタ，コピー機の大活躍だ。

お母さん業とおつとめでいそがしい委員一同は，１分でも早く帰宅できることを喜んでくれる。

さて，委員会の初仕事は恒例の「夏休み夜間パトロール」。全校936名の保護者全員が，夏休みのあいだ，夜の町をパトロールするのだ。このための，日程とチーム編成名簿をつくって，すべての保護者にくばらなければならない。

議案や作業プランはパソコンでやってきたけれど，委員みんなでやる名簿づくりは手仕事だ。まず，全校生徒から個人のデータをあつめる。住所，氏名，電話番号，当校在籍の兄弟の氏名を記入してもらい個人票をつくる。すべてこれによって名簿づくりが進行する。

通学区域には６つの町があって，パトロールも町単位で行われる。名簿は各町に人数ぶん必要になる。全校から集められた個人票は６つの町に分類されて，同じ家庭から通学する兄弟もひとつにまとめられる。

町ごとに，生徒数をパトロールの実施日数で割り，各班の人員を算出。住居表示を若い順から並べていき，日割りしながら当番の氏名を書きこんでいく。

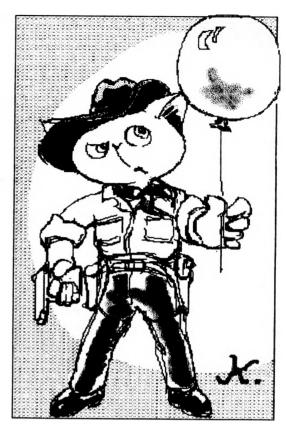
できあがった名簿はふたたび，各学年，各クラスに分けられて，生徒を通じて保護者の元に届けられる。

瀕死の淵から

７月初めまでに完成の予定で，名簿づくりのプロセスは始まった。

ところが，ひそかに予想したとおり，作業のじっさいはなかなか問題があった。まず，いちばんの根拠になる個人データが，生徒たちの手書きであること。こちらがワープロで準備した書式にも問題があったかもしれないが，文字の不明瞭，記入の不完全のために判読にまよった。中には欠席した友人のデータの代筆も多く，内容そのものの誤りもあった。

つぎに，住居表示が，町名以外はすべて数字であること。丁目，区画，マンション名，部屋番号が数字だけで識別される。そのためにわずかな数字のちがいが，大きな分類ちがいになる。また，この千に近い数字を若い順に並べる仕事は，やってみるととてもたいへんなのだ。

あんのじょう，できあがった名簿が保護者の元に届いたとたん，誤記や分類ちがいの苦情がひきもきらない。個人票の欠陥や書き写し時のミスなどなど，いろいろな原因だ。

こんなこと，パソコンを使えばパーフェクトに近いのに。学校の名簿を借りて個人のデータを入力すれば，あとは表集計のソフトがみんなやってくれる。

兄弟姉妹の一括，町の分類，住居表示の昇順整列，日分け，印刷。できあがりも手書きよりは上等だ。土曜日ごとに，必死の思いでかけつけて，あたふたと家路をたどる委員たちは，もっとラクになる。

と，ここまできて，ふと気がついた。ラクになって，なんにも仕事のないPTAこそ最悪なのだ。手間と時間のかかることをどんどん切り捨てて，合理化だけを提案したら，仮死状態のPTAがもう一度死んでしまう。だから，今のPTAにハイテクのススメはできない。

でも，ほんとにそうだろうか。あきらかに機械のほうがふさわしいとわかっていることを，いつまでも手作業でつづけるのが正しいのだろうか。

機械にまかせたら，あとに何も残らないPTAなら，一度絶命するのもいいかもしれない。

◆X-OVER NIGHT

(クロスオーバーナイト)

[第4話]

流行歌を追え!

TAKAHARA HIDEKI 高原 秀己

ぼくは比較的，歌謡曲もニューミュージックもポップスもと分けへだてなく幅広く聞くほうである。しかし，歌手の顔と名前が一致しないケースが極度に増えてきたのは困った状況だ。

最近，テレビの歌番組を見ることがめったにないことが大きいのだろう。コンサートにはもちろん行かない。FMラジオで聞いて，CDをレンタル店で借りて，気に入ったら買う，というパターンばっかりなので，「実物」を見る機会はほとんどない。これでは，歌と歌手の名前が一致しないのも無理がないところだ。

もちろんWinkはわかる。永井真理子もわかる。だがPRINCESS PRINCESSをバラバラにされると，かなり危なくなってくる。ティーンエイジャー歌手もダメ。そして何よりも，特に最近ミュージックシーンで頭角を現してきた新進バンドはわからない。

こうした人気バンドの歌はひと通りは聞いている。たまや米米CLUBはもちろん，BAKUFU-SLUMP, BEGIN, JITTERIN' JINN, X, LINDBERG……と最低限のヒットメーカーの歌は聞いているし，10曲くらいはカラオケで歌えるようにはしてある。

こうした歌を歌って，

「いまの歌は何？」

というようなオジサンは論外としても，結構知らない若手の同僚は多い。

もっとも，歌は知っていて歌えても，顔までははっきりわからないという人はぼくだけでもないようだ。それからいくと，ぼくは平均的なのかもしれない。

それにしても，最近のミュージックシーンはバンドの台頭により，激変しているといって間違いない。そもそもはアイドル偏重だったわけだが，バンドの登場により，アイドル歌手のヒットチャートにおける上位進出はここ半年で絶望的な状況となっている。人気がなければ，ゴールデンタイムあたりのテレビにはホイホイとは出られない。そのせいか，アイドル歌手の出番は午後5時以前に追いやられてしまった。それに歌番組自体が少なくなったこともある（深夜以外は)。

また，彼らはテレビ出演よりもCD販売とかコンサートを重視するので，ぼくたちは歌をテレビの歌番組で聞くことよりも，CDやコンサートで聞くほうが主体になってきたのだろう。

*　　*　　*

ところでバンド，特にアマチュアバンドは自然発生的にわき出てきたものなのだろうか？　もちろん「イカ天」などの舞台設定が整って，彼ら自身の登竜門ができたことは大きい。だが，それだけかというと，そうでもないようだ。

X-OVER NIGHTらしく，深夜の推論をしてみると……

ヒットメーカーでない場合，彼らはいわゆるフリーター，つまり定職は持たずにアルバイトで自由に時間が使えるようにして収入を確保する場合が多いという。

アルバイトで生活するなど，ちょっと前はかなり難しいことだった。それに耐えて今日がある，なんていう元劇団出身の俳優のサクセスストーリーがあるくらいだから，実態は想像できるというものだ。

ところが昨今の経済情勢を眺めてみると，アルバイトの賃金がこのところウナギ登り。普通のアルバイトでも東京では時給1,000円以上の求人がごろごろしているくらいだ。もちろん銀行，証券関係の会社員や待遇のいいSEなんかには及ばないものの，そこいらの製造業系中小企業なんかに比べると，ずっと収入はいい。アルバイトで食いつないでメジャーになるチャンスを待つことは十分に可能な時代となっている。

さらに世の中の雰囲気として，経済状況が見かけ上に過ぎないにせよ余裕が出てきているため，そうした生き方が許容されるようになってきたという変化も見逃すことはできない。髪の染め方やファッションが英国系ミュージックシーンの流れを追っていることと，英国風のいい意味で退廃した生活パターンとがダブってくるような気もしてしまう。

もうひとつ，これはこじつけっぽいのだが，東南アジアからやってくる苦学生や労働者が急増していることもあって，他人と自分を意識的に区別するために，ほかの先進国風の生活を追ってみるという機運があちこちで出てきていることも，見逃せないような気もする。

こんな感じで考えてくると，バンドの流行と経済情勢とは根の部分で因果関係としてつながりがあるような気がしてしまうのである。

まあ何にせよ，音楽業界自体が彼らの台頭もあって，アイドル偏重から本当に実力のある歌手を求めるように少しずつ変わってきたことはいいことなのだろう。

もっとも現在の形が完成形であるとは，とても考えられない。この鍵はメディアの変革によって，かなり左右されてくるとぼくは思う。そもそもCD, LDの普及によって，音楽の楽しみ方自体が変わったことはいうまでもない。

ニューメディアブームのお先棒を担ぐわけではないが，これからはCATV，衛星放送がより普及するだろうし，ハイビジョンも出てくる。そしてパソコンも，いま考えられている形ではないだろうが，メディアの端末機として活躍の場ができてくるはず。

はたして音楽とパソコンとの未来的な関係とは……。

THE SENTINEL

●グラフィックの統一環境

S-OS企画がスタートし，同じZ80を搭載したマシンならMZだろうがX1だろうが，はたまたPC-8801だろうが同じオブジェクトプログラムが動くようになってしばらくたった頃の話です。グラフィックも統一できるのではないかという話が持ち上がりました。S-OSの系譜（12）でも少し触れましたが，かくしてグラフィックの統一というかつてない試みがなされたのが1986年９月号で発表されたMAGICです。

コマンドを解釈実行するインタプリタ型のグラフィックルーチンながら，驚異的な描画スピードで3Dもなんのその。当時の編集部では3Dがはやっていたこともあり，青赤眼鏡を掛け正四面体をグルグル回すなどという遊びが流行しました。

当然機種別のプログラムですので，S-OS用のアプリケーションとしては発表されず，共通化の別の試みとして単体で供給されました。しかしこのままではマシン語から使うしかないためS-OS上の高級言語がグラフィックルーチンとしてサポートするようになりました。いまではS-OSのひとつの環境としてなくてはならない存在です。

第98部
BILLIARDS

●あなたもハスラー

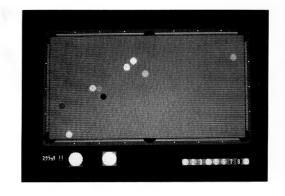
今月お届けするBILLIARDSは，このMAGICとSOROBANを利用したSLANGのアプリケーション。グラフィックをフルに使って，色とりどりの玉がグリーンの台に美しく表示される本格的なビリヤードゲームです。

動き始めると白い点になってしまうのはご愛敬。高速化のための工夫でしょう。10個の玉が動き始めると，動体の衝突という時間のかかる計算を延々と繰り返すのですからZ80にはかなり重い仕事です（Z80にもコプロセッサがあればなぁ）。でも総じて市販のビリヤードゲーム並によくできていますので，楽しめることでしょう。

編集部ではバンキングの代わりにジャンケンで先攻後攻を決め，対戦９ボールが盛り上がっています。皆さんも友達どうしワイワイいいながら挑戦してみてください。

さて，来月はサブルーチンをソースファイルではなくアセンブルして蓄えておくためのツール，リロケータブルファイルには不可欠なライブラリアンの登場です。

●S-OSの系譜(13)

1986年11月号に掲載されたエレショウのレポートは16ビットのX1，X68000の興奮のレポートで

始まっています。斬新なマンハッタンシェイプ。65536色のグラフィック。ソフトがないなどと，現在の状況からは想像もできないような中傷がなされたパソコン誌があったことを思い出します。

この11月号のSENTINELではゲームが２つ発表されました。ひとつはHOTTANというパズルゲーム，もうひとつはMAZE in MAZEというロールプレイング風の3D迷路ゲームです。

HOTTANはX1，MZで大好評を博したパズルゲームPITMANによく似たゲームでした。土の中に埋められた宝を，土を掘りながら集めていく単純なゲームなのですが，土の中には岩があり，岩の下の土を掘ると上から岩が落ちてきます。また，空中を動くことはできないためところどころ設置された梯をうまく使いながら宝を集めていかなければならないというものです。

最近ではこういったアクション型のパズルゲームが少なくなってしまったような気がします。アイデア勝負で作れるタイプのゲームですから，ひねった投稿に期待したいところです。もう一方のMAZE in MAZEはキャラクタを使って3D画面を構成した，気軽に楽しめる迷路ゲームでした。

読者の皆さんのなかにも情報処理試験を受けようと思っている方もいらっしゃるかと思います。第２種情報処理試験のアセンブリ言語の試験にはCOMP-Xという仮想CPU用のアセンブラであるCAP-Xが使用されてきました。S-OSでは1985年10月号でCAP-Xを発表しています。

しかし，COMP-Xはあまりに実用的でないとの批判から，1987年４月実施の第２種情報処理試験以降はCOMETという仮想CPU用のアセンブラCASLに変更されることとなりました。1986年12月号ではこのCOMETのシミュレータとCASLが登場しています。

CASLはZEDAと同様に使用できるエディタアセンブラで，得られたオブジェクトをCOMETシミュレータで実行させることができます。受験を考えている方はバックナンバーを引っぱり出して試してみてはいかがでしょう。

全機種共通S-OS“SWORD”′要

BILLIARDS

SLANG, SOROBAN, そしてMAGIC。共通システムとして発表された多くの優秀なパッケージ群を集大成して，グラフィック対応のビリヤードゲームが登場です。多くの要素をシミュレートしたリアルなゲーム感覚でハスラー気分を味わってください。

Kaneko Isamu
金子　勇

SLANGとMAGICを使ったビリヤードゲームを作ったので発表します。S-OS対応といってもなめてはいけません。回転も計算しており，押し球，引き球，ひねり，カーブボール，マッセなどもできます。S-OSと関連システムだけでもここまでできるというところを見てください。

なお，このゲームを動かすためにはSLANGのほかにMAGICとSOROBANが必要です。また，残念ながらSLANGのオンメモリ版ではメモリが足りないのでコンパイルできなくなってしまいました。

入力方法

まず，グラフィックライブラリのソースをお手持ちのエディタで入力し,GRAPHICS.LIBの名前でセーブしてください。

次にビリヤードプログラムのソースリストを入力してセーブします。そのあとコンパイルするわけですが，そのままではハッシュテーブルが足りないのでSLANGの3012Hを80H，3013Hを01Hにしてからコンパイルします。なお，コンパイルする前にSOROBANをNEW LOADERで9F00Hに組み込んでおいてください。

コンパイル中，SOROBAN.LIBとGRAPHICS.LIBをインクルードします。コンパイルが終わったら，

S BIL:5000:9569:5000:9000

としてセーブしてください。そのあと，SOROBANおよび各機種のMAGICとBILを読み込み，

S BIL:5000:C1FF:5000

のようにまとめてセーブします。次回からはこのBILを読み込むだけで実行できます。なお，実行時は25行スクロールモードにしてください。

操作法

このゲームはビリヤードの9ボールのシミュレーションです。球の打ち方から説明しましょう。まず，方向を決め，キューを持つ手のグリップの高さを決め，そのあとボールを打つ力の強さを決めると球が打てます。方向の選択は7と9で10°ずつ，4と6で1°，1と3で0.1°ずつ行うことができます。

方向が決まったらスペースを押します。ボールのどこを突くかは2，4，6，8で決め，グリップの高さは2，8で決めます。

このグリップの高さというのをなにに使うのかわからない人もいるでしょう。球の端を打つときグリップを上げておくと，そちら側に曲がるカーブボールにできるのです（ジャンプボールにはなりません）。

最後に打つ力の強さを決定します。マーカーが動くので好きなところでスペースを押してください。マーカーの位置は右側のほうが強くなります。

9ボールとは

9個の番号のついた的球とひとつの手球を使ってやるビリヤードのゲームです。

的球のうち番号のいちばん小さいものに当ててから的球のうちのどれかをポケットに落とします。うまく落とせたら同様にプレイを続けていき，ひとつも球が落ちなかったり，違う番号の球に最初に当たったりするともうひとりのプレイヤーと交代です（といっても，このゲームではコンピュータとの対戦はできませんから，対戦相手をどこかで調達しなければなりません)。目的は9番の球を落とすことです。それまでの経過に関わらず，先に9番の的球を落としたほうの勝ちとなります。細かいルールは省略します。人に聞くか，本を読んでください。

●マッセ

特別な打ち方としてマッセがあります。これはキューを立てて突く方法で，球をかなり曲げることができます。ただ，思いどおりに打つのは難しいでしょう。

マッセを使うには方向を決めたあとでMのキーを押しマッセのモードに入らなくてはなりません。ちなみに，このとき決める方向は自分の向き，つまりどの方向が正面になるかを表します。

次に上から見てどこを打つかを決め，グリップの位置を決めて（2を押すほどグリップが傾いたことになる）打つ力を指定します。以上でマッセができます。

特別なキー操作

通常のゲーム進行に関わりないモード切り替えのキーとしていくつかのキーが使用されています。

H　ヘルプモード切り替え

方向選択のときに押すと有効です。このモードでは手球と的球の当たる位置と当たったあとの手球と的球の進む方向を表示します。ただし，回転は考えていません。

R　リプレイ

これを押すと球の配置が1打前の位置に戻ります。

ESC･n　移動

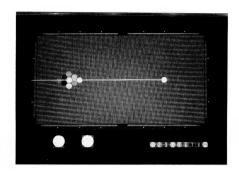
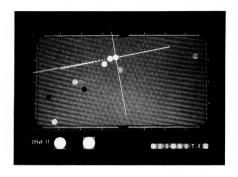
エスケープキー（ブレイクキー）を押して0～9のキーを押すと好きな球を移動させることができます。

そのほか，球が動いているときに1，2，3のキーを押すと表示モードを変更できま

す。1なら球を点で表し，跡が残ります。2では跡は残らないようになります。3ならば，ちゃんと色分けした丸い球を表示します。本当は3のモードで使いたいところですが，遅い(MAGICのCIRCLE FULLを使っている）ので速いのが好きな人は1か2を使いましょう。

プログラムについて

実行すればわかりますが，実行速度はあまり速くありません。これは表示にMAGICを使用し，実数演算を多用していることによります。MAGIC自体は高速かつ高機能なグラフィックパッケージですが，ここで必要とされているものに対応するにはMAGICが提供するものよりもっと単純な処理でいいわけです。演算についても実際には固定小数点演算でもできるのですが，試しにやってみたら誤差がひどいのでやめました。

このプログラムではできるだけ動きを本物に近づけるために球の回転まで考えているので実数演算でないとつらいのです。プログラム中では球とラシャの摩擦，球とクッションの摩擦，球同士の摩擦を計算しているので，球はかなり複雑な動きをします。たとえば，入射角と反射角が等しくないなどもありうるのです。

ただ，速度重視のために球の当たり判定で少し手を抜いてあるので球が3つ以上接触して並ぶと，思い切り打った手球がすり抜けたりすることもあります。ただし，ブレイクのときだけは別処理なのでご安心を。そのほか，ポケットの処理が結構いい加減です。角の処理はしていないので角クッションなどもできません。

＊　＊　＊

本当ならジャンプボールにも対応させたかったのですが遅くなるのと表示が難しくなるのでやめました。

S-OS対応に発表されたアプリケーションのなかでも結構凝ったゲームになっているのではないでしょうか。

SLANGはなかなか使いやすい言語でいいと思いますが実数を使うと大変なことになりますね。実数が手軽に扱える言語がS-OS用にほしいものです。

リスト1 GRAPHICS.LIB

```
  1 // graphics liblaly
  2
  3  array byte co[14];
  4  var   tile1,tile2;
  5
  6 init()
  7   begin
  8     call($af00);
  9     window(0,0,639,199);
 10     palet(1,2,3,4,5,6,7);
 11     mode(2,2); cls();
 12     mode(2,1); cls();
 13     mode(2,0); cls();
 14   end;
 15
 16 mode(mode,plane)
 17   begin
 18     co[0]=7;
 19     co[1]=mode;
 20     co[2]=plane;
 21     co[3]=15;
 22     ^ix=&co;
 23     call($b004);
 24   end;
 25
 26 cls()
 27   begin
 28     co[0]=9;
 29     co[1]=15;
 30     ^ix=&co;
 31     call($b004);
 32   end;
 33
 34 window(minx,miny,maxx,maxy)
 35   begin
 36     co[0]=6;
 37     memw[&co+1]=minx;
 38     memw[&co+3]=miny;
 39     memw[&co+5]=maxx;
 40     memw[&co+7]=maxy;
 41     co[9]=15;
 42     ^ix=&co;
 43     call($b004);
 44   end;
 45
 46 palet(a1,a2,a3,a4,a5,a6,a7)
 47   begin
 48     co[0]=10;
 49     co[1]=0;
 50     co[2]=a1;
 51     co[3]=a2;
 52     co[4]=a3;
 53     co[5]=a4;
 54     co[6]=a5;
 55     co[7]=a6;
 56     co[8]=a7;
 57     co[9]=15;
 58     ^ix=&co;
 59     call($b004);
 60   end;
 61
 62 line(x1,y1,x2,y2)
 63   begin
 64     co[0]=0;
 65     co[1]=2;
 66     memw[&co+2]=x1;
 67     memw[&co+4]=y1;
 68     memw[&co+6]=x2;
 69     memw[&co+8]=y2;
 70     co[10]=15;
 71     ^ix=&co;
 72     call($b004);
 73   end;
 74
 75 box(x1,y1,x2,y2)
 76   begin
 77     co[0]=2;
 78     memw[&co+1]=x1;
 79     memw[&co+3]=y1;
 80     memw[&co+5]=x2;
 81     memw[&co+7]=y2;
 82     co[9]=15;
 83     ^ix=&co;
 84     call($b004);
 85   end;
 86
 87 circle(x,y,r)
 88   begin
 89     co[0]=5;
 90     memw[&co+1]=tile1;
 91     memw[&co+3]=tile2;
 92     memw[&co+5]=x;
 93     memw[&co+7]=y;
 94     memw[&co+9]=r;
 95     co[11]=15;
 96     ^ix=&co;
 97     call($b004);
 98   end;
 99
100 full(x1,y1,x2,y2)
101   begin
102     co[0]=4;
103     memw[&co+1]=tile1;
104     memw[&co+3]=tile2;
105     memw[&co+5]=x1;
106     memw[&co+7]=y1;
107     memw[&co+9]=x2;
108     memw[&co+11]=y2;
109     memw[&co+13]=15;
110     ^ix=&co;
111     call($b004);
112   end;
113
114 mask(t1,t2)
115   begin
116     tile1=t1;
117     tile2=t2;
118   end;
119
```

リスト2 BILLIARDS

```
  1 //
  2 //     BILLIARDS  1990/4/19
  3 //
  4 //   Programed by Isamu Kaneko
  5 //
  6
  7  org    $5000;
  8  offset $4000;
  9  work   $d000;
 10
 11 #include SOROBAN.LIB
 12 #include GRAPHIC.LIB
 13
 14  const   w=9,
 15          p1=300,
 16          p2=p1*2;
 17
 18  var     a,b,bk,f2,f3,f4,ff,fm,hl,hl2,k,l,rf2,
 19          x2,x4,xxx,y2,y4,yyy,pf,sp;
 20
 21  array   !c1[4],!c2[4],!c3[4],!c4[4],
 22          !dt1[4],!dt4[4],!dt10[4],!dt100[4],!dt1000[4],
 23          !x[9][4],!vx[9][4],!xx[9][4],!x1[4],!x3[4],!dwx[4],!rx[9][4],
 24          !y[9][4],!vy[9][4],!yy[9][4],!y1[4],!y3[4],!dwy[4],!ry[9][4],
 25          !ww[9][4],!s1[9][4],!s2[9][4],
 26          !dm[4],!dm2[4],!t[4],!r[4],!tt[4],!rr[4],!dw[4],!z[4],
 27          !f[9],!f1[9],!f5[9],!rf[9],!vram[80*44];
 28
 29 main()
 30  begin
 31   @single();
 32   pf=0; hl=0;
 33   repeat [
 34     width(80); init(); palet(1,2,3,4,7,6,7);
 35     for a=0 to 80*44 vram[a]=255;
 36     f2=1; bk=1;
 37     @cvitf(t,80);
 38     @cvitf(c1,50);  @cvitf(c2,10);
 39     @cvitf(c3,100); @cvitf(c4,2000);
 40     @cvitf(dt1,1);
 41     @cvitf(dt4,4);
 42     @cvitf(dt10,10);
 43     @cvitf(dt100,100);
 44     @cvitf(dt1000,1000);
 45     for a=0 to w  f[a]=1;
 46     @cvitf(x[0],235); @cvitf(y[0],85);
 47     @cvitf(x[1],85 ); @cvitf(y[1],85);
 48     @cvitf(x[2],67 ); @cvitf(y[2],75);
 49     @cvitf(x[3],49 ); @cvitf(y[3],85);
 50     @cvitf(x[4],67 ); @cvitf(y[4],95);
 51     @cvitf(x[5],76 ); @cvitf(y[5],80);
 52     @cvitf(x[6],76 ); @cvitf(y[6],90);
 53     @cvitf(x[7],58 ); @cvitf(y[7],80);
 54     @cvitf(x[8],58 ); @cvitf(y[8],90);
 55     @cvitf(x[9],67 ); @cvitf(y[9],85);
 56     for a=0 to w [
 57       @move(rx[a],x[a]);
 58       @move(ry[a],y[a]);
 59       rf[a]=f[a];
 60     ]
 61     rf2=1;
 62
 63     mask($aa55,$aa55);
 64     mode(3,1);
 65     full(11,0,626,6);
 66     full(626,6,639,163);
 67     full(11,163,626,169);
 68     full(0,10,11,163);
 69     circle(11,6,12);
 70     circle(626,6,12);
 71     circle(11,163,12);
 72     circle(626,163,12);
 73     mode(3,2);
 74     full(418,191,630,199);
 75     full(11,6,626,163);
 76     full(80,198,120,199);
 77     full(180,179,220,199);
 78     mask($ffff,$ffff);
 79     mode(0,2);
 80     box(18,9,619,160);
 81     box(17,9,620,160);
 82     for k=0 to 2 [
 83       mode(3,k);
 84       for a=0 to 6 [
 85         if a!=3 [
 86           full(a*75+95,3,a*75+96,3);
 87           full(a*75+95,166,a*75+96,166);
 88         ]
 89       ]
 90       for a=0 to 2 [
 91         full(6,a*37+48,7,a*37+48);
 92         full(633,a*37+48,634,a*37+48);
 93       ]
 94     ]
 95     mask(0,0);
 96     for k=1 to 2 [
 97       mode(3,k);
 98       circle(22,12,18);
 99       circle(614,12,18);
100       circle(22,157,18);
101       circle(614,157,18);
102       circle(320,12,20);
103       circle(320,157,20);
104     ]
105     mode(3,2);
106     mask($aa55,$aa55);
107     full(19,10,618,159);
108     mask($ffff,$ffff);
109     for k=0 to 2 [
110       mode(3,k);
111       circle(100,189,20);
112       circle(200,189,20);
113     ]
114
115     repeat [
116       mode(3,0); mask(0,0);
117       full(19,10,618,159);
118       mask($ffff,$ffff);
119       for a=w downto 0 [
120         if f[a]>=1 [
121           ball(@cvfti(xx[a]),@cvfti(yy[a]),255,1);
122         ]
123       ]
124       for a=w downto 0 [
125         f1[a]=0;
126         xxx=@cvfti(x[a]); yyy=@cvfti(y[a]);
127         @cvitf(vx[a],0); @cvitf(vy[a],0);
128         @cvitf(ww[a],0); @cvitf(s1[a],0); @cvitf(s2[a],0);
129         if f[a]>=1 [
130           if xxx.<.15  @cvitf(x[a],15);
131           if xxx.>.304 @cvitf(x[a],304);
132           if yyy.<.15  @cvitf(y[a],15);
133           if yyy.>.154 @cvitf(y[a],154);
134           ball(@cvfti(x[a]),@cvfti(y[a]),a,1);
135         ]
136       ]
137       ball_number();
138
139       xxx=1800; yyy=0; f3=0;
140       mask($ffff,$ffff);
141       repeat [
142         @cvitf(dm,10);
143         @cvitf(x3,xxx); @div(x3,x3,dm); @rad(x3,x3);
144         @sin(y3,x3); @cos(x3,x3);
145         @cvitf(dm,300); @mul(x3,x3,dm); @mul(y3,y3,dm);
146         @add(x3,x3,x); @add(y3,y3,y); @cvitf(dm,2); @mul(x3,x3,dm);
147         x2=@cvfti(x)*2; y2=@cvfti(y); x4=@cvfti(x3); y4=@cvfti(y3);
148         mode(1,0);
149         line(x2,y2,x4,y4);
150         hl2=0;
151         if hl [
152           @cvitf(dm,10);
153           @cvitf(x3,xxx); @div(x3,x3,dm); @rad(x3,x3);
154           @sin(y3,x3); @cos(x3,x3);
155           @mul(x3,x3,dt10); @mul(y3,y3,dt10);
156           @move(vx,x3); @move(vy,y3);
157           @mul(dm,vx,vx); @mul(dm2,vy,vy);
158           @add(dm,dm,dm2); @sqr(z,dm);
159           @sub(x1,x,x[f2]);
160           @sub(y1,y,y[f2]);
161           @mul(dm,y1,vx);
162           @mul(dm2,x1,vy);
163           @sub(dm,dm,dm2); @div(rr,dm,z);
164           @abs(dm2,rr);
165           if @cmp(dm2,dt10)!=1 [
166             @mul(dm2,rr,rr); @sub(dm2,dt100,dm2);
167             @sqr(dm2,dm2); @neg(tt,dm2);
168             @mul(dm,tt,vx); @mul(dm2,rr,vy);
169             @sub(dm,dm,dm2); @div(dm,dm,z);
170             @add(dwx,x[f2],dm);
171             @mul(dm,rr,vx); @mul(dm2,tt,vy);
172             @add(dm,dm,dm2); @div(dm,dm,z);
173             @add(dwy,y[f2],dm);
174             @sub(x1,x[f2],dwx);
175             @sub(y1,y[f2],dwy);
176             @mul(x1,x1,dt10);
177             @mul(y1,y1,dt10);
178             mask($ffff,$ffff);
179             help_print();
180             hl2=1;
181           ]
182         ]
183         k=inkey(1);
184         mode(1,0);
185         line(x2,y2,x4,y4);
186         if hl2 [
187           mask(0,0);
188           help_print();
189         ]
190         case k [
191           27        : ball_move(1);
192           '1'       : xxx--;
193           '3'       : xxx++;
194           '4'       : xxx=xxx-10;
195           '6'       : xxx=xxx+10;
196           '7'       : xxx=xxx-100;
197           '9'       : xxx=xxx+100;
198           'H' , 'h' : [
199             if hl then hl=0; else hl=1;
200           ]
201           'R' , 'r' : [
202             for a=w downto 0 [
203               if f[a]>=1 [
204                 ball(@cvfti(x[a]),@cvfti(y[a]),255,1);
205               ]
206             ]
207             for a=w downto 0 [
208               if rf[a]>=1 [
209                 @move(x[a],rx[a]);
210                 @move(y[a],ry[a]);
211                 ball(@cvfti(x[a]),@cvfti(y[a]),a,1);
212                 f[a]=rf[a];
```

▶「創刊号を3名の方にプレゼント」するそうだが，10年後にそれを10万円（税別）で売るというやつが「ぼくらの掲示板」に登場しないか心配である。　神生総一(24)北海道

```
213             ]
214           ]
215           if bk==2 bk=1;
216           f2=rf2;
217           ball_number();
218         ]
219       ]
220     ] until ( k==' ' or k=='M' or k=='m' );
221     repeat ; until inkey(0)==0;
222     if k==' ' fm=0; else fm=1;
223     x2=7; y2=7;
224     repeat [
225       mode(1,2);
226       line(76+x2*2+fm*100,182+y2,96+x2*2+fm*100,182+y2);
227       line(86+x2*2+fm*100,177+y2,86+x2*2+fm*100,187+y2);
228       k=inkey(1);
229       line(76+x2*2+fm*100,182+y2,96+x2*2+fm*100,182+y2);
230       line(86+x2*2+fm*100,177+y2,86+x2*2+fm*100,187+y2);
231       case k [
232         '4'       : if x2.>.0    x2--;
233         '6'       : if x2.<.14   x2++;
234         '8'       : if y2.>.fm*7 y2--;
235         '2'       : if y2.<.14   y2++;
236         'M' , 'm' : fm=1;
237       ]
238     ] until k==' ';
239     x2=x2-7; y2=y2-7;
240     while inkey(0)==' ' ;
241     if fm then sp=7; else sp=0;
242     repeat [
243       mode(1,2);
244       box(95+x2*2+fm*100,187+y2-sp+fm*7,105+x2*2+fm*100,191+y2-sp+fm*7);
245       k=inkey(1);
246       box(95+x2*2+fm*100,187+y2-sp+fm*7,105+x2*2+fm*100,191+y2-sp+fm*7);
247       case k [
248       '8' : if sp.<.7 sp++;
249       '2' : if sp.>.0 sp--;
250       ]
251     ] until k==' ';
252     while inkey(0)==' ' ;
253     ff=1;
254     repeat [
255       locate(yyy,22); print(" <> ");
256       for a=0 to p2 a=a;
257       yyy=yyy+ff;
258       if yyy>73 ff=-ff;
259     ] until inkey(0)==' ';
260       @cvitf(dm,10);
261       @cvitf(x3,xxx); @div(x3,x3,dm); @rad(x3,x3);
262       @sin(y3,x3); @cos(x3,x3);
263       @cvitf(dm,yyy); @mul(vx,x3,dm); @mul(vy,y3,dm);
264       @cvitf(dm,4);
265       @div(vx,vx,dm); @div(vy,vy,dm);
266     if fm [
267       sp=8-sp;
268       x2=-x2; y2=-y2;
269       @cvitf(dm,y2); @mul(s1,vx,dm); @cvitf(dm,x2);
270       @mul(dm,dm,vy); @add(s1,s1,dm); @mul(s1,s1,dt4);
271       @cvitf(dm,y2); @mul(s2,vy,dm); @cvitf(dm,x2);
272       @mul(dm,dm,vx); @sub(s2,s2,dm); @mul(s2,s2,dt4);
273       @div(vx,vx,dt10); @div(vy,vy,dt10);
274       @cvitf(dm,sp);
275       @mul(vx,vx,dm); @mul(vy,vy,dm);
276     ]
277     else [
278       sp++;
279       @mul(dm,vx,vx); @mul(dm2,vy,vy);
280       @add(dm,dm,dm2); @sqr(dm,dm);
281       x2=-x2;
282       @cvitf(ww,x2); @mul(ww,ww,dm);
283       @cvitf(dm,4); @div(ww,ww,dm);
284       @cvitf(dm2,10);
285       y2=-y2;
286       @cvitf(dm,y2); @mul(s1,vx,dm); @cvitf(dm,sp.*.x2);
287       @mul(dm,dm,vy); @div(dm,dm,dm2); @add(s1,s1,dm);
288       @cvitf(dm,y2); @mul(s2,vy,dm); @cvitf(dm,sp.*.x2);
289       @mul(dm,dm,vx); @div(dm,dm,dm2); @sub(s2,s2,dm);
290     ]
291     f[0]=2;
292     for a=0 to w [
293       f5[a]=0;
294       @move(rx[a],x[a]);
295       @move(ry[a],y[a]);
296       rf[a]=f[a];
297       @move(xx[a],x[a]); @move(yy[a],y[a]);
298     ]
299     if bk==2 bk=0;
300     rf2=f2;
301     ball(@cvfti(xx),@cvfti(yy),255,1);
302     locate(0,23); print("            ");
303     locate(yyy,22); print("    ");
304
305     repeat [
306       k=inkey(0);
307       case k [
308         '1' : pf=5;
309         '2' : pf=3;
310         '3' : pf=1;
311         $1b : [ locate(0,0); stop(); ]
312       ]
313       if pf>2 [
314         pf=pf-3;
315         for a=w downto 0 [
316           if f[a]==2 [
317             ball(@cvfti(xx[a]),@cvfti(yy[a]),255,1);
318           ]
319         ]
320       ]
321       for a=w downto 0 [
322         if f[a]==2 [
323           @add(x[a],x[a],vx[a]);
324           @add(y[a],y[a],vy[a]);
325           @sub(dw,vx[a],s1[a]);
326           @div(dm,dw,c2); @add(s1[a],s1[a],dm);
327           @div(dm,dw,c3); @sub(vx[a],vx[a],dm);
328           @div(dm,vx[a],t);
329           @sub(vx[a],vx[a],dm);
330           @sub(dw,vy[a],s2[a]);
331           @div(dm,dw,c2); @add(s2[a],s2[a],dm);
332           @div(dm,dw,c3); @sub(vy[a],vy[a],dm);
333           @div(dm,vy[a],t);
334           @sub(vy[a],vy[a],dm);
335           b=ball_check(@cvfti(x[a]),@cvfti(y[a]),a);
336           if ( b!=255 and f[b]>0 ) [
337             @sub(x1,vx[a],vx[b]);
338             @sub(y1,x[a],x[b]);
339             @mul(dm,x1,y1);
340             @sub(x1,vy[a],vy[b]);
341             @sub(y1,y[a],y[b]);
342             @mul(dm2,x1,y1);
343             @add(dm,dm,dm2);
344             @cvitf(dm2,0);
345             if ( @cmp(dm,dm2)!=1 or f5[b]==0 ) [
346               @sub(x1,x[a],x[b]);
347               @sub(y1,y[a],y[b]);
348               @mul(dm,x1,x1); @mul(dm2,y1,y1); @add(r,dm,dm2);
349               if ( f[a]==1 or f[b]==1 ) [
350                 @mul(dm,vx[a],vx[a]); @mul(dm2,vy[a],vy[a]);
351                 @add(dm,dm,dm2); @sqr(z,dm);
352                 @sub(x1,x[a],x[b]);
353                 @sub(y1,y[a],y[b]);
354                 @mul(dm,y1,vx[a]);
355                 @mul(dm2,x1,vy[a]);
356                 @sub(dm,dm,dm2); @div(rr,dm,z);
357                 @abs(dm2,rr);
358                 if @cmp(dm2,dt10)!=1 [
359                   @mul(dm2,rr,rr); @sub(dm2,dt100,dm2);
360                   @sqr(dm2,dm2); @neg(tt,dm2);
361                   @mul(dm,tt,vx[a]); @mul(dm2,rr,vy[a]);
362                   @sub(dm,dm,dm2); @div(dm,dm,z);
363                   @add(x[a],x[b],dm);
364                   @mul(dm,rr,vx[a]); @mul(dm2,tt,vy[a]);
365                   @add(dm,dm,dm2); @div(dm,dm,z);
366                   @add(y[a],y[b],dm);
367                   @cvitf(r,100);
368                   ball_collide();
369                 ]
370               ]
371               else [
372                 if @cmp(r,dt100)!=1 [
373                   ball_collide();
374                 ]
375               ]
376             ]
377           ]
378
379           ff=0;
380           xxx=@cvfti(x[a]); yyy=@cvfti(y[a]);
381           if xxx.<.15 [
382             @sub(dw,vy[a],ww[a]); @mul(dw,dw,vx[a]); @div(dw,dw,c1);
383             @sub(ww[a],ww[a],dw); @add(vy[a],vy[a],dw);
384             @cvitf(dm,30); @sub(x[a],dm,x[a]); @neg(vx[a],vx[a]);
385             ff=1;
386           ]
387           if xxx.>.304 [
388             @add(dw,vy[a],ww[a]); @mul(dw,dw,vx[a]); @div(dw,dw,c1);
389             @sub(ww[a],ww[a],dw); @sub(vy[a],vy[a],dw);
390             @cvitf(dm,608); @sub(x[a],dm,x[a]); @neg(vx[a],vx[a]);
391             ff=1;
392           ]
393           if yyy.<.15 [
394             @add(dw,vx[a],ww[a]); @mul(dw,dw,vy[a]); @div(dw,dw,c1);
395             @sub(ww[a],ww[a],dw); @add(vx[a],vx[a],dw);
396             @cvitf(dm,30); @sub(y[a],dm,y[a]); @neg(vy[a],vy[a]);
397             ff=1;
398           ]
399           if yyy.>.154 [
400             @sub(dw,vx[a],ww[a]); @mul(dw,dw,vy[a]); @div(dw,dw,c1);
401             @sub(ww[a],ww[a],dw); @sub(vx[a],vx[a],dw);
402             @cvitf(dm,308); @sub(y[a],dm,y[a]); @neg(vy[a],vy[a]);
403             ff=1;
404           ]
405           if ff==1 [
406             if ( xxx.<.25  and yyy.<.25  ) ff=2;
407             if ( xxx.>.295 and yyy.<.25  ) ff=2;
408             if ( xxx.>.295 and yyy.>.145 ) ff=2;
409             if ( xxx.<.25  and yyy.>.145 ) ff=2;
410             if ( xxx.>.150 and xxx.<.170 and yyy.<.50  ) ff=2;
411             if ( xxx.>.150 and xxx.<.170 and yyy.>.120 ) ff=2;
412             if ff==2 [
413               locate(0,23); print(a," in !");
414               f[a]=0; ball(@cvfti(xx[a]),@cvfti(yy[a]),255,1);
415               f1[a]=1;
416               ball_number();
417             ]
418           ]
419
420           if f[a] [
421             ball(@cvfti(xx[a]),@cvfti(yy[a]),255,pf);
422             ball(@cvfti(x[a]),@cvfti(y[a]),a,pf);
423             @move(xx[a],x[a]); @move(yy[a],y[a]);
424           ]
425         ]
426       ]
427
428       l=0;
429       for a=0 to w [
430         if f[a]==2 [
431           @abs(dm,vx[a]); @abs(dm2,vy[a]);
432           if ( @cmp(dm,dt1)==-1 and @cmp(dm2,dt1)==-1 ) [
433             @abs(dm,s1[a]); @abs(dm2,s2[a]);
434             if ( @cmp(dm,dt1)==-1 and @cmp(dm2,dt1)==-1 ) [
```

▶弟がパソコンを買うというので，教育的指導を行い，X68000PROIIに決めさせました。
山中毅(30)神奈川県

```
435                 f[a]=1;
436                 @cvitf(ww[a],0); @cvitf(s1[a],0); @cvitf(s2[a],0);
437                 @cvitf(vx[a],0); @cvitf(vy[a],0);
438                 ball(@cvfti(x[a]),@cvfti(y[a]),a,1);
439               ]
440             ]
441           ]
442           if f[a]==2 l=1;
443         ]
444       ] until l==0;
445
446       for a=w downto 0 [
447         if f[a]>=1 [
448           ball(@cvfti(x[a]),@cvfti(y[a]),a,1);
449         ]
450       ]
451       ff=0;
452       if f1[0]==1 [
453         locate(0,23); print("ｽｸﾗｯﾁ !!"); beep(); ff=1; f3=f2;
454       ]
455       if f2!=f3 [
456         locate(0,23); print("ﾌｧｰﾙ !!"); beep(); ff=1;
457       ]
458       if ff==1 [
459         xxx=85;
460         for a=1 to w [
461           if f1[a]==1 [
462             while vram[1680+xxx/4]!=255 xxx=xxx-10;
463             @cvitf(x[a],xxx); @cvitf(y[a],85);
464             f[a]=1; f1[a]=0; xxx=xxx-10;
465           ]
466         ]
467         ball_number();
468         scratch();
469       ]
470       f2=9;
471       for a=1 to w [
472         if ( f[a]>=1 and a<f2 ) f2=a;
473       ]
474       if f1[9]==1 [ ff=2; beep(); ]
475     ] until ff==2;
476   ] until 0;
477 end;
478
479 ball_collide()
480   begin
481     if ( f3==0 and a==0 ) f3=b;
482     if bk!=1 [
483       @sub(x1,x[a],x[b]);
484       @sub(y1,y[a],y[b]);
485       @sub(dm,vx[a],vx[b]); @sub(dm2,vy[a],vy[b]);
486       @mul(dm,dm,x1); @mul(dm2,dm2,y1);
487       @add(dm,dm,dm2); @div(dm,dm,r);
488       @mul(dm2,dm,y1); @mul(dm,dm,x1);
489       @add(vx[b],vx[b],dm); @add(vy[b],vy[b],dm2);
490       @sub(vx[a],vx[a],dm); @sub(vy[a],vy[a],dm2);
491       @add(dwx,ww[a],ww[b]); @mul(dwx,dwx,dm2); @div(dwx,dwx,c4);
492       @add(dwy,ww[a],ww[b]); @mul(dwx,dwx,dm);  @div(dwy,dwy,c4);
493       @add(vx[a],vx[a],dwx); @add(ww[a],ww[a],dwx);
494       @sub(vx[b],vx[b],dwx); @sub(ww[b],ww[b],dwx);
495       @sub(vy[a],vy[a],dwy); @add(ww[a],ww[a],dwy);
496       @add(vy[b],vy[b],dwy); @sub(ww[b],ww[b],dwy);
497       if f[b]==1 ball(@cvfti(xx[b]),@cvfti(yy[b]),255,1);
498       f[a]=2; f[b]=2; f5[a]=1; f5[b]=1;
499     ]
500     else  [
501       break(); bk=2;
502     ]
503   end;
504
505 ball(x,y,c,f)
506   begin
507     var xx,yy,a;
508     if ( x.>=.15 and x.<=.304 and y.>=.15 and y.<=.154 ) [
509       if f!=2
510         if f then ball_1(x,y,c);
511         else ball_2(x,y,c);
512       else
513         if c!=255 ball_2(x,y,c);
514       xx=x/4; yy=y/4;
515       vram[yy*80+xx]=c;
516     ]
517   end;
518
519 ball_1(x,y,c)
520   begin
521     x=x.*.2;
522     mode(3,0);
523     case c [
524       2,4,0,9 : mask($ffff,$ffff);
525       others  : mask(0,0);
526     ]
527     circle(x,y,9);
528     mode(3,1);
529     case c [
530       1,3,4,5,9,0 : mask($ffff,$ffff);
531       7           : mask($aa55,$aa55);
532       others      : mask(0,0);
533     ]
534     circle(x,y,9);
535     mode(3,2);
536     case c [
537       1,6,9,0 : mask($ffff,$ffff);
538       5,255   : mask($aa55,$aa55);
539       others  : mask(0,0);
540     ]
541     circle(x,y,9);
542     if c==9 [
543       mask(0,0); mode(3,0);
544       full(x-9,y-2,x+9,y+1);
545     ]
546   end;
547
548 ball_2(x,y,c)
549   begin
550     x=x.*.2;
551     if c==255 [
552       mode(3,0); mask(0,0);
553     ]
554     else [
555       mode(2,0);  mask($ffff,$ffff);
556     ]
557     full(x-2,y-1,x+2,y+1);
558   end;
559
560 ball_check(x,y,c)
561   begin
562     var xx,yy,b,f,d;
563     xx=x/4-2; yy=y/4-2;
564     b=yy*80+xx;
565     f=255;
566     for yy=0 to 4 [
567       for xx=0 to 4 [
568         d=vram[b];
569         if ( d!=255 and d!=c and d<f ) f=d;
570         b++;
571       ]
572       b=b+75;
573     ]
574   return(f);
575   end;
576
577 break()
578   begin
579     var a;
580     @move(dm,vx);
581     @abs(dm,dm);
582     for a=0 to w [
583       @cvitf(vx[a],rnd(800)-400);
584       @div(vx[a],vx[a],dt1000);
585       @mul(vx[a],vx[a],dm);
586       @cvitf(vy[a],rnd(800)-400);
587       @div(vy[a],vy[a],dt1000);
588       @mul(vy[a],vy[a],dm);
589     ]
590     for a=0 to w [
591       f[a]=2; ball(@cvfti(xx[a]),@cvfti(yy[a]),255,1);
592     ]
593   end;
594
595 scratch()
596   begin
597     mask($ffff,$ffff);
598     ball(@cvfti(xx),@cvfti(yy),255,1);
599     for a=w downto 1 [
600       if f[a]>=1 [
601         ball(@cvfti(xx[a]),@cvfti(yy[a]),255,1);
602       ]
603     ]
604     for a=w downto 1 [
605       if f[a]>=1 [
606         ball(@cvfti(x[a]),@cvfti(y[a]),a,1);
607       ]
608     ]
609     ball_move(0);
610     f[0]=1;
611   end;
612
613 ball_number()
614   begin
615     var a,c;
616     for a=1 to w [
617       c=a;
618       locate(50+a*3,24);
619       if f[a] [
620         print(a);
621       ]
622       else [
623         c=255; print(" ");
624       ]
625       ball_1(a*12+202,195,c);
626     ]
627   end;
628
629 ball_move(c)
630   begin
631     var kk,xxx,yyy,k,a;
632     if c [
633       repeat [
634         repeat [
635           k=inkey(1);
636           k=k-'0';
637         ] until k<10;
638       ] until f[k];
639       kk=k;
640     ]
641     else [
642       kk=0;
643       k=f2;
644     ]
645     mask($ffff,$ffff);
646     ball(@cvfti(x[kk]),@cvfti(y[kk]),255,1);
647     xxx=@cvfti(x[k]); yyy=@cvfti(y[k]);
648     repeat [
649       mode(1,0);
650       line(xxx*2-10,yyy,xxx*2+10,yyy);
651       line(xxx*2,yyy-5,xxx*2,yyy+5);
652       for a=0 to pl a=a;
653       line(xxx*2-10,yyy,xxx*2+10,yyy);
654       line(xxx*2,yyy-5,xxx*2,yyy+5);
655       k=inkey(0);
656       case k [
```

▶私もトマト（野菜のほうが好き）ジュースは大好きなんですよ。あの酸味がたまらないですなあ。コショウを振ってもうまいし。 富田祐樹(17)東京都

```
657           '4' : if xxx.>.15  xxx--;
658           '6' : if xxx.<.304 xxx++;
659           '8' : if yyy.>.15  yyy--;
660           '2' : if yyy.<.154 yyy++;
661         ]
662       ] until k==' ';
663       @cvitf(x[kk],xxx); @cvitf(y[kk],yyy);
664       ball(@cvfti(x[kk]),@cvfti(y[kk]),kk,1);
665       repeat ; until inkey(0)==0;
666     end;
667
668 help_print()
669     var x,y;
670     begin
671       x=@cvfti(dwx).*.2;
672       y=@cvfti(dwy);
673       mode(1,0);
674       @add(dm,x1,dwx);
675       @add(dm2,y1,dwy);
676       line(x,y,@cvfti(dm).*.2,@cvfti(dm2));
677       @sub(dm,dwx,x1);
678       @sub(dm2,dwy,y1);
679       line(x,y,@cvfti(dm).*.2,@cvfti(dm2));
680       @sub(dm,dwx,y1);
681       @add(dm2,dwy,x1);
682       line(x,y,@cvfti(dm).*.2,@cvfti(dm2));
683       @add(dm,dwx,y1);
684       @sub(dm2,dwy,x1);
685       line(x,y,@cvfti(dm).*.2,@cvfti(dm2));
686       mode(3,0);
687       circle(x,y,9);
688     end;
```

全機種共通システムインデックス

■85年6月号
- 序論　共通化の試み
- 第1部　S-OS"MACE"
- 第2部　Lisp-85インタプリタ
- 第3部　チェックサムプログラム

■85年7月号
- 第4部　マシン語プログラム開発入門
- 第5部　エディタアセンブラZEDA
- 第6部　デバッグツールZAID

■85年8月号
- 第7部　ゲーム開発パッケージBEMS
- 第8部　ソースジェネレータZING

■85年9月号
- インタラプト　S-OS番外地
- 第9部　マシン語入力ツールMACINTO-S
- 第10部　Lisp-85入門(1)

■85年10月号
- 第11部　仮想マシンCAP-X85
- 連載　Lisp-85入門(2)

■85年11月号
- 連載　Lisp-85入門(3)

■85年12月号
- 第12部　Prolog-85発表

■86年1月号
- 第13部　リロケータブルのお話
- 第14部　FM音源サウンドエディタ

■86年2月号
- 第15部　S-OS"SWORD"
- 第16部　Prolog-85入門(1)

■86年3月号
- 第17部　magiFORTH発表
- 連載　Prolog-85入門(2)

■86年4月号
- 第18部　思考ゲームJEWEL
- 第19部　LIFE GAME
- 連載　基礎からのmagiFORTH
- 連載　Prolog-85入門(3)

■86年5月号
- 第20部　スクリーンエディタE-MATE
- 連載　実戦演習magiFORTH

■86年6月号
- 第21部　Z80TRACER
- 第22部　magiFORTH TRACER
- 第23部　ディスクダンプ&エディタ
- 第24部　"SWORD" 2000 QD
- 連載　対話で学ぶ magiFORTH
- 特別付録　PC-8801版S-OS"SWORD"

■86年7月号
- 第25部　FM音源ミュージックシステム
- 付録　FM音源ボードの製作
- 連載　計算力アップのmagiFORTH
- 特別付録　SMC-777版 S-OS"SWORD"

■86年8月号
- 第26部　対局五目並べ
- 第27部　MZ-2500版 S-OS"SWORD"

■86年9月号
- 第28部　FuzzyBASIC 発表
- 連載　明日に向かって magiFORTH

■86年10月号
- 第29部　ちょっと便利な拡張プログラム
- 第30部　ディスクモニタ DREAM
- 第31部　FuzzyBASIC 料理法〈1〉

■86年11月号
- 第32部　パズルゲーム HOTTAN
- 第33部　MAZE in MAZE
- 連載　FuzzyBASIC 料理法〈2〉

■86年12月号
- 第34部　CASL & COMET
- 連載　FuzzyBASIC 料理法〈3〉

■87年1月号
- 第35部　マシン語入力ツールMACINTO-C
- 連載　FuzzyBASIC 料理法〈4〉

■87年2月号
- 第36部　アドベンチャーゲーム MARMALADE
- 第37部　テキアベ作成ツール CONTEX

■87年3月号
- 第38部　魔法使いはアニメがお好き
- 第39部　アニメーションツール MAGE
- 付録　"SWORD" 再掲載と MAGIC の標準化

■87年4月号
- 第40部　INVADER GAME
- 第41部　TANGERINE

■87年5月号
- 第42部　S-OS"SWORD" 変身セット
- 第43部　MZ-700用 "SWORD" を QD 対応に

■87年6月号
- インタラプト　コンパイラ物語
- 第44部　FuzzyBASIC コンパイラ
- 第45部　エディタアセンブラ ZEDA-3

■87年7月号
- 第46部　STORY MASTER

■87年8月号
- 第47部　パズルゲーム碁石拾い
- 第48部　漢字出力パッケージ JACKWRITE
- 特別付録　FM-7/77版 S-OS"SWORD"

■87年9月号
- 第49部　リロケータブル逆アセンブラ Inside-R
- 特別付録　PC-8001/8801 版 S-OS"SWORD"

■87年10月号
- 第50部　tiny CORE WARS
- 第51部　FuzzyBASIC コンパイラの拡張
- 第52部　X1turbo 版 S-OS"SWORD"

■87年11月号
- 序論　神話のなかのマイクロコンピュータ
- 付録　S-OS の仲間たち
- 第53部　もうひとつの FuzzyBASIC 入門
- 第54部　ファイルアロケータ&ローダ
- インタラプト　S-OS こちら集中治療室
- 第55部　BACK GAMMON

■87年12月号
- 第56部　タートルグラフィックパッケージTURTLE
- 第57部　X1turbo 版 "SWORD" アフターケア ラインプリントルーチン
- 特別付録　PASOPIA7 版 S-OS"SWORD"

■88年1月号
- 第58部　FuzzyBASIC コンパイラ・奥村版
- 付録　石上版コンパイラ拡張部の修正

■88年2月号
- 第59部　シューティングゲーム ELFES

■88年3月号
- 第60部　構造型コンパイラ言語 SLANG

■88年4月号
- 第61部　デバッギングツール TRADE
- 第62部　シミュレーションウォーゲーム WALRUS

■88年5月号
- 第63部　シューティングゲーム ELFES II
- 第64部　地底最大の作戦

■88年6月号
- 第65部　構造化言語 SLANG 入門(1)
- 第66部　Lisp-85 用 NAMPA シミュレーション

■88年7月号
- 第67部　マルチウィンドウドライバ MW-1
- 連載　構造化言語 SLANG 入門(2)

■88年8月号
- 第68部　マルチウィンドウエディタ WINER

■88年9月号
- 第69部　超小型エディタ TED-750
- 第70部　アフターケア WINER の拡張

■88年10月号
- 第71部　SLANG 用ファイル入出力ライブラリ
- 第72部　シューティングゲーム MANKAI

■88年11月号
- 第73部　シューティングゲーム ELFES Ⅳ

■88年12月号
- 第74部　ソースジェネレータ SOURCERY

■89年1月号
- 第75部　パズルゲーム LAST ONE
- 第76部　ブロックゲーム FLICK

■89年2月号
- 第77部　高速エディタアセンブラ REDA
- 特別付録　X1版 S-OS"SWORD"〈再掲載〉

■89年3月号
- 第78部　Z80用浮動小数点演算パッケージSOROBAN

■89年4月号
- 第79部　SLANG 用実数演算ライブラリ

■89年5月号
- 第80部　ソースジェネレータ RING

■89年6月号
- 第81部　超小型コンパイラTTC

■89年7月号
- 第82部　TTC用パズルゲーム　TICBAN

■89年8月号
- 第83部　CP/M用ファイルコンバータ

■89年9月号
- 第84部　生物進化シミュレーションBUGS

■89年10月号
- 第85部　小型インタプリタ言語TTI

■89年11月号
- 第86部　TTI用パズルゲーム　PUSH BON!

■89年12月号
- 第87部　SLANG用リダイレクションライブラリ DIO LIB

■90年1月号
- 第88部　SLANG用ゲームWORM KUN
- 特別付録　再掲載SLANGコンパイラ

■90年2月号
- 第89部　超小型コンパイラTTC++

■90年3月号
- 第90部　超多機能アセンブラOHM-Z80

■90年4月号
- 第91部　ファジィコンピュータシミュレーションI-MY

■90年5月号
- 第92部　インタプリタ言語STACK

■90年6月号
- 第93部　リロケータブルフォーマットの取り決め
- 第94部　STACK用ゲーム　SQUASH!
- 第95部　X68000対応S-OS"SWORD"
- 特別付録　PC-286対応S-OS"SWORD"

■90年7月号
- 第96部　リロケータブルアセンブラWZD

■90年8月号
- 第97部　リンカWLK

＊以上のアプリケーションは，基本システムである S-OS "MACE" または S-OS"SWORD" がないと動作しませんのでご注意ください。

▶いつもパソコン雑誌の中では“これしかない！”と思っています。柿島俊一(23)千葉県

マシン語カクテル in Z80's Bar
第15回――ハッシュでチェック――

シナリオ&イラスト：山田純二
特別監修：金子俊一&浦川博之
コンサルタント：中野修一

ソフトバンクの社屋移転に伴いかどうかは知りませんが，Z80's Barも新装開店だそうです。とはいえ，別に内容に変わりがあるわけでもなく，いつもどおりに話は進むのでした。ちなみに今月はハッシュ法とかいうものを使ったデータの検索のお勉強です。

新装開店新台多数

♪カラン，コロ～ン（ドアの開く音）
源光（以下光）：こんにちは～。
マスター（以下M）：やあ，こんにちは（ガサゴソ）。
ようこ（以下Yo）：あらどうしたの，光君。こんな昼間から。
光：Z80's Barが新装開店したというからご挨拶申し上げようと思いまして。だけど，すごい騒ぎだなあ。
Yo：だってまだ片付けも終わってないし，ひまなんだったら光君も手伝ってよ。
純二（以下純）：24051（プチ）24052（プチ）……。
光：わあ。なんだ純二君。そんなところにいたのか。何をやっているんだ，君は。
Yo：見てのとおり，エアーキャップ（通称プチプチ）を潰してるのよ。
光：こんなにひまな奴がいるんだったら，手伝わせればいいのに。
Yo：それがね，プチプチを見つけてからずうーっと，あれにかかりっきりで離れないのよ。
M：光君，ちょっとこっちで探し物を手伝ってもらえるかい。
光：はいはい，何を探してるんですか。
M：君が溜めているツケの明細書。
光：あ，急用ができたので，いずれまた。
Yo：うそだってば，光君。前に来たお客さんのプログラムが，どこかへ行ってしまったみたいなのよ。
光：なんだ，びっくりさせないでくださいよ。だけど，プログラムぐらい，店のコンピュータにでもぶちこんで管理すればいいのに。
M：そんなプログラムがあったら，とっくにそうしていますよ。
光：そんないいかげんな経営でよく潰れないなあ。ここはZ80's Barでしょ。プログラムなんて適当にツケの溜まっている客に作らせちゃったらどうです。
M：そうかそうか。そうすると光君がプログラムを作ってくれるんだね。
光：へ？
Yo：さっすが，光君。がんばってね。
光：ちょっと待って。なんでそっちのほうに話が飛ぶんですか。
M：やっぱりいいだしっぺがやるというのは民主主義の大原則（？）であるから。
光：そんなあ。
Yo：あたしからもお願いするから。はい，ホワイトチョコレートソーダあげる。
光：しょうがないなあ，ようこさんの頼みとあっちゃあ。
純：24111，この女ったらし（プチ）。
光：やかましい！

探し物見つけます

光：ところでマスター，どんなものがいいんですか？
M：とりあえず，データの検索ルーチンかなんか。
光：検索ねえ。ここのプログラムの量からすると，ハッシュ法を使ったほうがよさそうだなあ。
Yo：ねえねえ，データのチェックって最初のほうから順番に調べていけばいいんじゃないの？
光：基本的にはね。100件ぐらいのデータなら頭から「これじゃない，これでもない」とやっていけばいいけど，データの量が増えると最後のほうにあるデータを検索したいときにすごく時間がかかっちゃう。そこでハッシュ法を使うんだ。
Yo：ふ～ん。
光：ハッシュ法ではデータがどこに格納されているかを示すハッシュテーブルを用意することによって，検索の効率を上げることができるんだよ。たとえば，Oh!Xの目次からZ80's Barを探すとする。目次を順番に「Z80's，Z80's……」と探すのは記事が多い本になればなるほど大変でしょ？
Yo：うん。
光：ハッシュ法ってのは，たとえばZ80's Barに1番なら1番と番号を振って，探したいときに作っておいた表の1番のところを見れば「何ページ（にあるよ）」と書いてあるような方式だと思って。
Yo：あー，なるほど。そのページ一覧表みたいなのがハッシュテーブルなのね。でも，それはどうやって作るの？
光：まず，データを登録するときにハッシュ関数というものにデータを通して，ハッシュ値を求める。次に，ハッシュテーブルの中のハッシュ値の指定したところに，データがあるアドレスを登録してやるんだ（図1参照）。
M：でも光君，もしも違うデータが同じハ

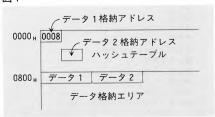
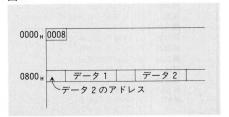



図1

図2

ッシュ値を示したらどうするんです。場所がダブッたらまずいでしょ。
光：そういう場合における処理方法の違いによってオープンハッシュ，チェーンハッシュという2つの方法に分けられます。
Yo：それって，どこがどう違うの。
光：オープンハッシュのほうは強制的に次のテーブルアドレスをそのデータのハッシュ値としてしまうんだ。
M：つまり，あるデータのハッシュ値は0000Hなんだけど，そこにはすでにアドレスが登録済みだった場合には，次の0002Hをデータのハッシュ値としてしまうんですね。
光：そのとおり。0002Hがだめだったら，その次の0004Hといった具合に，空きが見つかるまで探していくんですよ。
Yo：じゃあ,チェーンハッシュのほうは？
光：チェーンハッシュのほうはすでに登録されたデータの後ろにどんどん繋げていくんだ。登録するデータにはデータ同士の繋がりを示す2バイトのポインタを付加してやる必要があるけどね。
Yo：どういうこと？
光：具体的にいうと，まずハッシュ値（0000H）にデータ1のアドレス（1000H）が登録されているとする。そこに,ハッシュ値が同じであるデータ2が出てきたときには，

1．0000Hのアドレスを見にいく。
2．すでに，1000Hのアドレスがあるからデータ1を見にいく。
3．次に続いているデータはないのでポインタにデータ2のアドレスをセットして登録が終了する。

という具合に処理をしていくんです（図2参照）。
Yo：ということは，さらに同じハッシュ値のデータ3が出てきたときには，データ2のポインタにデータ3のアドレスをセットしてやればいいのね。
光：そういうこと。
純：25891（プチ）。データがデータにどんどんぶら下がっていくわけだね。

見つかるかな？

光：さて，今度はデータ検索について説明していきましょう。といっても，データの登録の様子を説明してきた時点である程度は想像がついただろうけど。まずは，オープンハッシュ。手順としては，

1．ハッシュ値を求める。
2．ハッシュテーブルに格納されているアドレスを取り出す。
3．もしもアドレスが登録されていなければ，探そうとしているデータは存在しないからエラーを表示してメインルーチンに戻る。
4．取り出したアドレスのデータと検索したいデータを比べていく。
5．あっていたらデータを取り出して，メインルーチンに戻る。
6．違っていたらハッシュ値をひとつずらして2から繰り返す。

というふうにしていく。
Yo：チェーンハッシュの場合は？
光：これもだいたい似たような感じで，

1～5．オープンハッシュと同じ処理。
6．違っていたらポインタのアドレスを取り出す。
7．次のデータが存在しなかったら探そうとしているデータは存在しないので，エラーを表示してメインルーチンに戻る。
8．次のデータのアドレスをセットして，4から繰り返す。

というふうにするんだ。
Yo：どっちのほうが効率がいいのかな。
光：ようこちゃんは，どっちだと思う。
Yo：やっぱり，処理の少ないオープンハッシュのほうだと思う。
光：マスターは？
M：私も，ようこさんと同じ……に，しようかと思ったけれど，芸がないのでチェーンハッシュにしましょう。
純：30302（プチ），僕はようこさんと同じだよ。
光：誰も貴様には聞いておらん。それにマスター，クイズじゃないんですから，わざわざ違う答えにすることはないでしょう。
M：いいじゃないの。
Yo：で，光君。正解はどっちなの。
光：僕はチェーンハッシュのほうだと思っているんだ。
M：我々が納得できる理由を述べよ。
光：それはですね。オープンハッシュの場合ハッシュテーブルがいっぱいになってくるにつれて登録，そして検索に時間がかかってしまうからですよ。極端にいうと，1024個のテーブルが用意されているけど，最後の1個しかテーブルの空きがなくて，最後に登録しようとしたデータのハッシュ値が空きの1つ後ろだった場合を考えてみてごらん。
Yo：1つ前に戻れば問題ないけど戻ることはできないから，結局1023回もチェックしていかなければならないんだ。
光：そのとおり。チェーンハッシュなら，いくらなんでも1000個ものデータが連なっていることはまずないといっていいからね。それに登録できるデータの数もオープンハッシュではハッシュテーブルの大きさに限定されてしまうけど，チェーンハッシュではそういうことがない。納得しました？
Yo&M：納得しました。
純：30374（プチ），うんうん。

プログラムの解説

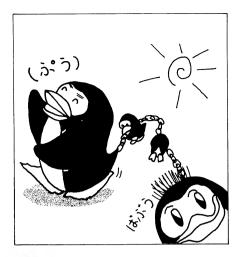
コロコロコロ～♪（電話が鳴る音）
純：はい，山田です。
光：あっ，山田君？　光だけど，今月ひましてる？
純：一応，ひまですけど。
光：それじゃあ，今月のプログラム，君が作ってくれない？　いま試験中で忙しくって。
純：へっ？
光：じゃあ，よろしくね（ガチャン）。
純：おーい，ひかるくんや～い。
電話：ツーツー……。

というわけで，身勝手な光君に代わって，作ったプログラムの解説をしていきます。

・HASHSEARCH
入力　DEレジスタ……ラベルネームを格納してあるアドレス
出力　HLレジスタ……データ
ハッシュテーブルからデータを取り出すサブルーチンです。

・HASHSET
入力　DEレジスタ……ラベルネームを格納してあるアドレス
(INPDAT)　……データ
出力　なし
ハッシュテーブルにラベルを登録するサブルーチン。まずハッシュ値を求めてから，すでに登録してあるか判定して，登録してあったらCHAINに飛んでいます。CHAINではラベルの二重定義チェックをしているところに注目。定義できるラベルは英数字のみです。

・HASH
入力　DEレジスタ……ラベルネームの格納アドレス
出力　HLレジスタ……ハッシュ値
ラベルのハッシュ値を計算するサブルーチン。このルーチンはREDAのものをそのまま流用してます。計算方法は文字ごとに値を5倍してキャラクタコードを加えて，さらに2倍していく。これを文字列が終わるまで繰り返しています。

・SEARCH
入力　HLレジスタ……ハッシュテーブル上のラベルデータ格納アドレス
出力　Cy=1　……一致していない
　　　Cy=0　……一致している
ハッシュテーブルにあるラベルネームと，登録しようとしているラベルネームが一致しているか調べます。

どこに使うんだ

Yo：ところで，ハッシュ法の原理はわかったけど，実際どんなところでこのハッシュ法が使われているの。

光：身近なところで，アセンブラやコンパイラのラベルチェックなんかに使われているよ。

Yo：どんなふうに？

光：アセンブラのREDAでは，1パス目に文法チェックとラベルテーブルの作成をやっているんだ。そのラベルテーブルの作成にオープンハッシュ法が使われていて，2パス目にラベルが使われている命令のときに，このハッシュテーブルを参照してアドレスをセットしていきながら，オブジェクトを作成していくんだ。そこで，アセンブラ(REDA)では2パスあるけど，コンパイラには1パスでオブジェクトを作成してしまうものがある(SLANGなど)。これはどうやっているのかわかるかな？

Yo&M：はーい光君，わかりませーん。

光：はいはい，単純明快な答えをどうもありがとう。んで，この場合はセットするアドレスをチェーンさせることによってラベルに対応するアドレスをセットしていくんですよ。これも具体的に説明していきましょう。登録するラベルデータは図3のような構成をとっているとしよう。たとえば，

```
#HAJIME
        LD   A,10
        DEC  A
        JP   NZ,#HAJIME
```

というプログラムではラベル(#HAJIME)があらかじめ登録されているので，JP命令がきたときにはハッシュテーブルを参照すれば素直に#HAJIMEのアドレスが取り出せるのですが，次のようなプログラムのときにはちょっと困ったことになるんだ。

```
CD D0 1F #HAJIME CALL #GETKEY
FE 05             CP   "5"
CA *1 *1          JP   Z,#JMP
05                DEC  B
CA *2 *2          JP   Z,#JMP
                  JR   #HAJIME
        #JMP
```

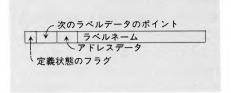
#GETKEYはあらかじめ登録されているとして，問題はラベル(#JMP)を登録する前にJP命令で#JMPが使われていること。ラベルが現れた，ということでコンパイラはハッシュテーブルにラベル(#JMP)を探しに行くわけですが，当然のことながらラベルは見つからない。そこで，とりあえず#JMPは未定義のフラグを立てて，ラベルアドレスをセットするところには＊1のアドレスをセットして登録します。このとき，＊1のアドレスにはとりあえず0000_Hをセットしておく。

次に5行目にも#JMPが使われているけど，このときには一応テーブルに存在しているが，まだアドレスは確定していない。このときには＊1のアドレスに＊2のアドレスをセット，＊2のアドレスには0000_Hをセットしておく。そして，7行目にしてやっとこさ#JMPがでてくるので，コンパイラはハッシュテーブルにラベルを定義しにいくんだな。そいでコンパイラはここで#JMPは未定義だ，ということを発見して＊1，＊2と#JMPのアドレスをセットしてまわり，未定義のフラグを消してアドレスもセットして，めでたしめでたし，となるんだ。

Yo&M：パチパチ～。

光：はあはあ。一気に説明したら息が切れちゃったけど，わかってくれました？

図3

Yo&M：ニコニコニコ……。

光：だめだこりゃ。

さあって来月は？

M：光君，どうもごくろうさま。あとはプログラムの作成だけだね。

光：はいはい，まかしておいてくださいよ。

♪カラン，コロ～ン(ドアの開く音)

老＆善＆で：こんにちは～。

Yo：あら，みんなおそろいでどうしちゃったの。

で：ひまだったもので。

善：そういうこと。あっ山田君，面白いことやってるね，僕にもやらせてよ。

純：40001(プチ)，いいよ。

純＆善：40002(プチ)，40003(プチ)……。

老：やれやれ，まだまだ子供じゃのう。ん，光君はずいぶんと疲れているようじゃの。

光：はい。

M：きょうはずいぶんとこき使ってしまいましたからね。

Yo：光君，さっきあげたホワイトチョコレートソーダまだ飲んでないじゃない。

光：そういえばそうだね。それじゃあみんなも一緒に乾杯しましょう。

M：そうですね。今日は私のおごりで。

で：さっすがマスター。ほら，山田君も西川君もこっちにきなよ。

純＆善：は～い。

M：それじゃあ，引っ越しの片付けはみんなでやるとして，とりあえずかんぱ～い。

一同：かんぱ～い。ごくごく……うっ。

純：鼻から血がでて，はなぢぶ～。

つづく

リスト1

```
0000                1
0000                2
0000                3  ; Z80'Bar CHAIN HASH SEARCH
0000                4  ;        1990.7.8  J.YAMADA
0000                5
1FE8 P              6  #MSG     EQU  $1FE8
1FE5 P              7  #MSX     EQU  $1FE5
1F94 P              8  #PEEK    EQU  $1F94
1F9A P              9  #POKE    EQU  $1F9A
1F97 P             10  #POKE@   EQU  $1F97
0000               11
0000               12
A000               13           ORG    $A000
A000               14
A000               15  ;HASH TABLE KARA DATA WO GET
A000               16  ; in  DE=LABEL NAME ADRESS
A000               17  ; out HL=DATA
A000               18
A000               19  HASHSEARCH
A000 CD BF A0      20           CALL   HASH
A003 CD 3B A0      21           CALL   HASHCUT
A006 CD 94 1F      22           CALL   #PEEK
A009 4F            23           LD     C,A
A00A 23            24           INC    HL
A00B CD 94 1F      25           CALL   #PEEK
A00E 47            26           LD     B,A
A00F B1            27           OR     C
A010 CA 13 A1      28           JP     Z,ERROR1     ;UNDEFINED ERROR
A013               29  SEA2
A013 60            30           LD     H,B
A014 69            31           LD     L,C
A015 CD F8 A0      32           CALL   SEARCH
```

```
A018 30 13            33         JR    NC,FIND
A01A CD 94 1F         34         CALL  #PEEK
A01D B7               35         OR    A
A01E CA 13 A1         36         JP    Z,ERROR1
A021 23               37         INC   HL
A022 CD 94 1F         38         CALL  #PEEK
A025 4F               39         LD    C,A
A026 23               40         INC   HL
A027 CD 94 1F         41         CALL  #PEEK
A02A 47               42         LD    B,A
A02B 18 E6            43         JR    SEA2
A02D                  44 FIND
A02D 23               45         INC   HL
A02E 23               46         INC   HL
A02F 23               47         INC   HL
A030 CD 94 1F         48         CALL  #PEEK
A033 5F               49         LD    E,A
A034 23               50         INC   HL
A035 CD 94 1F         51         CALL  #PEEK
A038 67               52         LD    H,A
A039 6B               53         LD    L,E
A03A C9               54         RET
A03B                  55
A03B                  56 ;HASU SIZE
A03B                  57
A03B                  58 HASHCUT
A03B 7C               59         LD    A,H          ;HASH SIZE $07FF
A03C E6 07            60         AND   $07
A03E 67               61         LD    H,A
A03F C9               62         RET
A040                  63
A040                  64 ;HASH TABLE NI TOUROKU
A040                  65 ;  in  DE=LABEL NAME ADRESS
A040                  66
A040                  67 HASHSET
A040 CD BF A0         68         CALL  HASH
A043 08               69         EX    AF,AF'
A044 3C               70         INC   A
A045 08               71         EX    AF,AF'
A046 CD 3B A0         72         CALL  HASHCUT
A049 CD 94 1F         73         CALL  #PEEK
A04C 4F               74         LD    C,A
A04D 23               75         INC   HL
A04E CD 94 1F         76         CALL  #PEEK
A051 47               77         LD    B,A
A052 B1               78         OR    C
A053 20 3B            79         JR    NZ,CHAIN     ;SUDENI TOUROKUZUMI
A055                  80
A055 2B               81         DEC   HL
A056 ED 5B 49 A1      82         LD    DE,(MSTRP)
A05A 7B               83         LD    A,E
A05B CD 9A 1F         84         CALL  #POKE
A05E 23               85         INC   HL
A05F 7A               86         LD    A,D
A060 CD 9A 1F         87         CALL  #POKE
A063                  88 HSS3
A063 97               89         SUB   A
A064 32 4B A1         90         LD    (STRDATA),A
A067 21 00 00         91         LD    HL,0000
A06A 22 4C A1         92         LD    (STRDATA+1),HL
A06D 2A 47 A1         93         LD    HL,(INPDAT)
A070 22 4E A1         94         LD    (STRDATA+3),HL
A073                  95
A073 01 05 00         96         LD    BC,0005      ;DATA SIZE
A076 08               97         EX    AF,AF'
A077 81               98         ADD   A,C
A078 4F               99         LD    C,A
A079 30 01           100         JR    NC,HSS2
A07B 04              101         INC   B
A07C                 102 HSS2
A07C 21 4B A1        103         LD    HL,STRDATA
A07F ED 5B 49 A1     104         LD    DE,(MSTRP)
A083 C5              105         PUSH  BC
A084 D5              106         PUSH  DE
A085 CD 97 1F        107         CALL  #POKE@
A088 E1              108         POP   HL
A089 C1              109         POP   BC
A08A 09              110         ADD   HL,BC
A08B 22 49 A1        111         LD    (MSTRP),HL
A08E B7              112         OR    A
A08F C9              113         RET
A090                 114
A090                 115 CHAIN
A090 60              116         LD    H,B
A091 69              117         LD    L,C
A092 CD F8 A0        118         CALL  SEARCH
A095 D2 18 A1        119         JP    NC,TWOERR    ;TWO STRINGS ERROR
A098 CD 94 1F        120         CALL  #PEEK
A09B B7              121         OR    A
A09C 28 0C           122         JR    Z,CHAIN2
A09E 23              123         INC   HL
A09F CD 94 1F        124         CALL  #PEEK
A0A2 23              125         INC   HL
A0A3 4F              126         LD    C,A
A0A4 CD 94 1F        127         CALL  #PEEK
A0A7 47              128         LD    B,A
A0A8 18 E6           129         JR    CHAIN
A0AA                 130 CHAIN2
A0AA 2F              131         CPL
A0AB CD 9A 1F        132         CALL  #POKE
A0AE 23              133         INC   HL
A0AF ED 5B 49 A1     134         LD    DE,(MSTRP)
A0B3 7B              135         LD    A,E          ;NEXT POINTER SET
A0B4 CD 9A 1F        136         CALL  #POKE
A0B7 23              137         INC   HL
A0B8 7A              138         LD    A,D
A0B9 CD 9A 1F        139         CALL  #POKE
A0BC C3 63 A0        140         JP    HSS3
A0BF                 141
A0BF                 142 ; CALUCULATE HASH
A0BF                 143 ;    in  DE=STRING ADRESS
```

```
A0BF                 144 ;  out  HL=HASH,A=LENGTH
A0BF                 145
A0BF                 146 HASH
A0BF 21 00 00        147         LD    HL,0000
A0C2 D9              148         EXX
A0C3 21 50 A1        149         LD    HL,STRING
A0C6 D9              150         EXX
A0C7                 151
A0C7 08              152         EX    AF,AF'
A0C8 97              153         SUB   A
A0C9                 154 HASH2
A0C9 08              155         EX    AF,AF'
A0CA CD E9 A0        156         CALL  LETCHECK
A0CD 38 14           157         JR    C,HASHRET
A0CF                 158
A0CF D9              159         EXX
A0D0 77              160         LD    (HL),A
A0D1 23              161         INC   HL
A0D2 D9              162         EXX
A0D3                 163
A0D3 4D              164         LD    C,L
A0D4 44              165         LD    B,H
A0D5 29              166         ADD   HL,HL
A0D6 29              167         ADD   HL,HL
A0D7 09              168         ADD   HL,BC
A0D8 4F              169         LD    C,A
A0D9 06 00           170         LD    B,00
A0DB 09              171         ADD   HL,BC
A0DC 29              172         ADD   HL,HL
A0DD 13              173         INC   DE
A0DE 08              174         EX    AF,AF'
A0DF 3C              175         INC   A
A0E0 20 E7           176         JR    NZ,HASH2
A0E2 C9              177         RET
A0E3                 178 HASHRET
A0E3 D9              179         EXX
A0E4 3E 0D           180         LD    A,$0D
A0E6 77              181         LD    (HL),A
A0E7 D9              182         EXX
A0E8 C9              183         RET
A0E9                 184
A0E9                 185 LETCHECK
A0E9 1A              186         LD    A,(DE)
A0EA FE 30           187         CP    "0"
A0EC D8              188         RET   C
A0ED FE 3A           189         CP    "9"+1
A0EF 3F              190         CCF
A0F0 D0              191         RET   NC
A0F1 FE 41           192         CP    "A"
A0F3 D8              193         RET   C
A0F4 FE 5B           194         CP    "Z"+1
A0F6 3F              195         CCF
A0F7 C9              196         RET
A0F8                 197
A0F8                 198 ;STRING SEARCH
A0F8                 199 ;  in  HL=DATA TOP
A0F8                 200
A0F8                 201 SEARCH
A0F8 E5              202         PUSH  HL
A0F9 01 05 00        203         LD    BC,0005
A0FC 09              204         ADD   HL,BC
A0FD 11 50 A1        205         LD    DE,STRING
A100                 206 SCH2
A100 CD 94 1F        207         CALL  #PEEK
A103 47              208         LD    B,A
A104 1A              209         LD    A,(DE)
A105 B8              210         CP    B
A106 20 08           211         JR    NZ,NO!
A108 FE 0D           212         CP    $0D
A10A 28 05           213         JR    Z,OK
A10C                 214
A10C 23              215         INC   HL
A10D 13              216         INC   DE
A10E 18 F0           217         JR    SCH2
A110                 218 NO!
A110 37              219         SCF
A111                 220 OK
A111 E1              221         POP   HL
A112 C9              222         RET
A113                 223
A113                 224 ;ERROR
A113                 225
A113                 226 ERROR1
A113 11 34 A1        227         LD    DE,NOTERRM
A116 18 03           228         JR    ER2
A118                 229 TWOERR
A118 11 20 A1        230         LD    DE,TWOERRM
A11B                 231 ER2
A11B CD E5 1F        232         CALL  #MSX
A11E 37              233         SCF
A11F C9              234         RET
A120                 235
A120 54 57 4F 20     236 TWOERRM DM    "TWO STRING ERROR!!"
A124 53 54 52 49
A128 4E 47 20 45
A12C 52 52 4F 52
A130 21 21
A132 0D 00           237         DB    $0D,00
A134 55 4E 44 49     238 NOTERRM DM    "UNDIFINED ERROR!!"
A138 46 49 4E 45
A13C 44 20 45 52
A140 52 4F 52 21
A144 21
A145 0D 00           239         DB    $0D,00
A147 00 00           240 INPDAT  DW    0000         ;INPUT DATA
A149 00 08           241 MSTRP   DW    $0800        ;STRING SET POINTER
A14B 00              242 STRDATA DB    00           ;FLAG
A14C 00 00           243         DW    0000         ;NEXT POINTER
A14E 00 00           244         DW    0000         ;DATA
A150                 245 STRING  DS    80           ;STRING BAFA
A1A0                 246
```

PC-E500テーブルトークRPGサポートシステム(2)

マスター戦闘支援ツールCST

Matsui Shin
松井 信

ポケコンを使ってテーブルトークRPGを楽しむための秘密兵器。いよいよ今回はそのプログラムを掲載し，その使われ方をゲーム進行に即して実践的にご紹介しましょう。その名もマスター戦闘支援ツールCST。どうですカッコイイでしょう。

まず，プログラムリストを見てください。これがPC-E500(PC-1480U/1490U)用，マスター戦闘支援ツール CST (Combat Support Tool)です。

前回，テーブルトークRPGには「マスターが大変である」という欠点があると述べました。AD&DおよびD&Dは，数あるテーブルトークRPGのなかでもかなり戦闘がスピーディなほうですが，それでも結構めんどいこともあります。

たとえば，D&Dにはトロールというやつがいます。これはもともとは北欧の妖精で，ムーミンのようないいやつもいるのですが，D&Dのは指輪物語に出てくる悪いやつです。このモンスターは，リジェネレート(時間の経過とともにダメージが回復する)し，また炎によるダメージはリジェネレートしないので別につけておく必要があり，さらに両手と口で3回攻撃するなど，マスターに負担を強いるモンスターなのです。

というわけで，戦闘の手順のうち，自動化できるところは自動化してしまおう，どうでもいい雑魚はコンピュータにやらせよう，というのが戦闘支援ツールCSTの考え方です。このプログラムはAD&D(2nd edition)用なのですが，D&Dへの移植は簡単にできるのでご安心を。

キャラクターの属性

とりあえず，AD&D(およびD&D)をやったことのない人のために，戦闘のルールを軽く説明しておきます。

まず，すべてのキャラクターおよびモンスターには，AC(アーマークラス。小さいほど敵の攻撃が当たりにくい)，HP(ヒットポイント。体力を表す。0になると死ぬ)，THAC0(AC0の相手に20面のサイコロの目でいくつ以上で当たるか。小さいほど攻撃がよく当たる)，#AT(攻撃回数)，damage(攻撃が当たったときに敵に与えるダメージ，普通「3d6+4」などと書く。ちなみにこれは，3個の6面サイコロの合計+4という意味)などの属性が設定されています。ほかに，魔法などの属性もあります。

また，これらの数値は，キャラクターのレベルやモンスターのHD(ヒットダイスという。モンスターの強さを表す)，および持っているアイテムなどにより決定されます。詳しくはAD&DまたはD&Dのマニュアルを見てください。

戦闘シーケンス

実際の戦闘例を見てみましょう。

アリーナ (レベル7ファイター，AC0，HP35，THAC013，#AT2，damage1d8+2)

ブライ (レベル6魔法使い，AC6，HP25，THAC019，#AT1，damage1d6)

の2人が洞窟にはいると，4匹のオーガ

Ogre(AC5，HD4+1，HP4d8+1(平均19)，#AT1，damage1d10)

それに，2匹のトロール

Troll(AC3，HD6+6，HP6d8+6(平均33)，#AT3，damage 1d4+4，1d4+4，2d6，戦闘開始後3ランド目から1ラウンド3HPのリジェネレート)

が現れた。まだ距離がある！　ブライの呪文が響く。「ファイアーボール！」洞窟に爆発の炎が広がった。

ここで，ブライをやっているプレイヤーはファイアーボールのダメージを決定します。というわけで，彼は6面を6個振り，23を出しました。

すかさずマスターはオーガとトロールの魔法回避のチェック(Saving Throwという)をします。20面を振って，ある数字以上ならばダメージが半分になります。そして，ダメージを現在のHPから引くわけです。

ファイアーボールに生き残ったトロールが2人に向かってくる。

マスターとプレイヤーはサイコロによってイニシアチブを取り合い，どちらから先に攻撃するか決定します。(コロコロ)，プレイヤーが先手を取りました。

アリーナの攻撃！　必殺の剣がトロールに向かう。

アリーナのプレイヤーは，20面を振って当たるかどうかを決定します。当たり！2回目の攻撃は……当たり！　ダメージを計算し，マスターに向かって「トロールAに15ダメージ！」などといい，マスターはトロールAのHPからダメージを引きます。

そして，トロールの反撃！

マスターはやはり20面を振り，当たったかどうかを計算します。トロールの場合，3回攻撃するわけです。当たったらダメージを計算します。そして「アリーナに6ダメージ！」などといい，次にトロールBの攻撃をします。

そして，すべて終わったら次のラウンドです。

CSTの使い方

最初は面倒臭いものですが，やがてルールは体に吸収され，ほとんど機械的にこなせるようになるものです。とはいうものの，マスターはすべてのモンスターを動かすわけですから大変なのは大変です。ではここでCSTを使うとどうなるでしょうか。

まず，CSTを起動するとファイル名を聞いてきますがとりあえず無視してリターンキーを押してください。すると，メイン画面になります。

```
GROUP0
GROUP1
GROUP2
S,A,R,0-2,Q>_
```

4行目にカーソルがあるはずです。この行をコマンドラインと呼びます。

●モンスターのセット

まず，

S0(ret)　　　(ret)はリターンキー

と入力してください。これはグループ0に

データをセットするという意味です。するとまず,

N=__

と聞いてくるので,モンスターの数を入力します。とりあえず,オーガ4匹なので,

N=4(ret)

とすると,次に,

HP=(/ret)__

と聞いてきます。HPが決まっている場合はそれを入力します。ここではHPはHDから自動計算させることにして,(ret)とします。すると,

HD/LV=?

と表示されるので,オーガのHDである,4+1(ret),を入力します。ここで,敵がモンスターでなく人間の場合は,F(fighter),C(cleric),T(thief),M(magic user)のアルファベットに続けてレベルを入力します。「ex. F5(ret)」。

また,決まったHPを入力したときは,

HD/LV=(ret)

と表示されるので,もしそのモンスターが攻撃をしないとき(つまり,後述のHPcalcだけを使いたい場合)は,(ret)とすればこれ以降をパスします。

次に,攻撃のデータを聞いてきます。

#at=__

オーガは1回しか攻撃しないので,1(ret)と答えます。そして,

dmg 1 (/N)=?

に対し,オーガのダメージ1d10を,

dmg 1 (/N)=1*10

(ndm+pは,n*m+pと書く),または,

dmg 1 (/N)=1−10

と入力してください(1から10という意味)。複数攻撃する場合,間違えたら,N,を押すとひとつ前に戻ります。最後に,

TH adj=__

と出てきますが,オーガには当たりやすさ(To Hit)の修正はないので,(ret)とします。すると,それぞれのHPやTHAC0が自動的に計算され,画面は,

```
GROUP0 A13 B17 C23 D23
GROUP1
GROUP2
S,A,R,0-2,Q>_
```

などとなるはずです。A~Dのアルファベットはグループ内でのモンスターの識別記号で,そのすぐあとの数字はそれぞれのHPです(乱数によって決まるのでだいたいこのぐらい)。とりあえずグループ0にオーガのデータセットが完了しました。次に,グループ1に,トロールをセットします。

S1(ret)	:グループ1をセット
2(ret)	:2匹
(ret)	:HPはコンピュータが決定
6+6(ret)	:HD=6+6
3(ret)	:#AT=3
5−8(ret)	:damage 5−8(or 1*4+4)
(ret)	:同じ場合は省略できる
2*6(ret)	:damage 2−12 (or 2*6)
(ret)	:当たりやすさの修正なし

これで,トロールがセットされました。

ところで,仮にアリーナをグループ2にセットするとしたら,

S2(ret)	:グループ2
1(ret)	:1人
35(ret)	:HP=35
F7(ret)	:Level 7 Fighter
2(ret)	:2回攻撃
1*8+2(ret)	:ダメージ1d8+2
(ret)	:同じなので省略
1	:剣の当たりやすさに+1の修正がある。

となります。

モンスターのグループは0~2の3つで,それぞれ8匹まで。それぞれのモンスターの攻撃回数は8回までとなっています。

なお,すでに使われているグループを使おうとすると,OK?(Y/N),と聞いてくるので,YかNで答えてください。ここで,「N=」と数を聞いてきたとき,(ret)とすると,そのグループは初期化されます(何も設定していないことになる)。

また,グループ間コピーということができます。メイン画面で,

s2=1(ret)

などとすると,モンスターの数を聞いてきたあと,グループ1のモンスターデータがグループ2にコピーされます。

●モンスターへの攻撃

メイン画面で,グループ番号だけを入力するとそのグループのHPcalcモードに入ります。

とりあえず,ブライのファイアーボールということで,まずメイン画面で,0(ret)とすると,コマンドラインが,

HP calc (0):__

となるはずです。これがHPcalcモードで,この段階ではグループ0のダメージを計算できます。オーガAからDにファイアーボールで23ダメージということで,

HP calc (0):ABCD23F/16(ret)

と入力してください。これは,ABCDに,炎によって(F),23ダメージ,Saving Throwは16,という意味です。そして,(ret)とすると,HPcalcモードから抜けるので,次に,1(ret)として,HPcalcをグループ1に移し,トロールのダメージを計算します。

HP calc (1):AB23F/14(ret)

次にアリーナの与えたダメージを入力。

HP calc (1):A15(ret)

剣のダメージにはSaving Throwはないので,このようになります。

また,HPcalcモードに入らずに,メイン画面から直接,

S,A,R,0−2,Q>0ABCD23F/16(ret)

のようにしてもダメージを入力できます。このときはモードに入らずにメイン画面に戻ってきます。

ところで,ファイアーボールのところで触れましたが,ダメージを記述するとき後ろに「F」をつけると炎によるダメージということになります。これはリジェネレートするモンスターのためのものです。

また,特殊なダメージとして,魔法などによるマヒや金縛りがあります。これは,ダメージのところを「*」で置き換えてください。Saving Throwも(設定していたら)実行されます。たとえば,

HP calc (1):AB*/14(ret)

のようにします。マヒしたモンスターは,アルファベットが小文字になり,攻撃できなくなります。マヒを解除するには,もう一度「*」をすれば戻ります。

●モンスター側の攻撃

それでは,モンスターの攻撃です。メイン画面から,Aに続けて攻撃するグループの番号を入力します。この場合,グループ1のトロールならば,

S,A,R,0−2,Q>A1(ret)

ゲーム紹介(2)

Advanced Dungeons & Dragons

TSR inc.

AD&DはD&D(Dungeons & Dragons)の改良版である。英米ではD&Dよりもはるかに主流で,おそらく世界でいちばんメジャーなテーブルトークRPGだろう。D&Dの手軽さ(プレイアビリティの高さ)をできるだけ生かしながら,発散していくゲームバランスを抑え込む努力が各所に見られる。

現在は第2版 (2nd edition)が出回り始めたところであり,日本語版も徐々に訳が出る予定だ。また,英語が読めるならば膨大な数の良質な市販シナリオを利用できる。私が思うに最もよくできたテーブルトークRPGである。

マスターガイド,プレイヤーハンドブック,モンスターコンペディウムが最低必要で,ほかに拡張ルールや世界設定集などありとあらゆるものがある。みんな揃えようと思うととってもお金がかかる。

とします。すると画面は，

```
G1 d20[14][7][15]
th0 13  −1  6  −2
dmg   6  6   4
−>A      hit ret/N!
```

のようになります（乱数によって決まるので必ずしもこのとおりではない）。

左側上から，「G1」はグループ1，「th0 13」は，THAC0が13である，「−>A」は，トロールAの攻撃である，ということを表しています。また，「d20」の右側は，攻撃のサイの目，その下はそのサイの目で当たるAC，その下，「dmg」の右側は，それぞれの攻撃のダメージです。

ここで，(ret)とすると，そのグループの次のモンスターに移ります。また，Nを入力すると，直接メイン画面に戻ります。なおここでは，サイの目の「20」は絶対当たり，「1」は絶対外れとしています。

●リジェネレート

モンスターのリジェネレートもサポートしています。リジェネレートが始まるラウンドになったら，メイン画面から，

Rn m(ret)

とすると，グループnに，毎ラウンドmHPのリジェネレートがセットされます。たとえば，グループ1，毎ラウンド3HPなら，

R1 3(ret)

とします。セットが終わったら，毎ラウンドの最初にメイン画面で，

R(ret)

とすれば，自動的にすべてのリジェネレートが行われます。また，炎によるダメージ(Fをつけたダメージ）はリジェネレートしませんし，最初のHPを上回ることはありません。

●ファイル入出力

CSTを終了するには，メイン画面から，

Q(ret)

とします。すると，

quit : filename/ret/N＝＿

と表示されます。ここで，(ret)とすると，終了します。また，なにかファイルネームを指定してやると，現在のデータをファイルに書き出してから，終了します。また，Nとするとメイン画面に戻ります。

ここで書き出したデータファイルは，RAMファイルに記憶されています。CSTの起動時に，このファイルネームを指定すると，そのデータファイルを読み込みます。

したがって，その日に出てくるモンスターをあらかじめセットしておくことにより，そのときの手間を省くことができます。

●実行の仕方

プログラムは7.5Kバイトありますからそれ以上のRAMファイルを確保してから，リストを入力してください。PC−E500のようにエンジニアリングモードが使える機種では，ファイル名を「COMBAT.1##」などとしておくとエンジニアリングモードで呼び出せて便利です。実際，私のE500ではメニューはすべてAD&D用に使われています。すべてBASICなのでデバッグは楽ですが，最初のほうに「on error goto」が使ってあるので，デバッグが終わるまではREM文にしておいてください。

開発に当たって

このプログラムはもともとPC−E500で開発したものですが，今回PC−9801上のエディタで書き直し，RS−232CでE500に転送したものです（E500のRAMファイルが飛んでしまって最初のプログラムが消えてしまったからです）。

結局98上で再度開発したのですが，開発はパソコンのほうが遥かに効率が上がるし安全性も向上します。本格的にポケコンを使いたい人には，RS−232Cレベルコンバータ「CE−140T」でパソコン（テキストファイルがエディットできるもの）に接続することをおすすめします。

なお，このCE−140Tのマニュアルに掲載されているサンプルの通信プログラムはかなり使い勝手が悪く，なんだこりゃ？と思って使っていたのですが，あるときふと，

save "com:"
load "com:"

とすればいいということに気づきました。普通の人はそうやって使うんだから，そのぐらいマニュアルに書いておいてくれればよかったのに。といいつつ以後は便利に使っています。通信速度も9600bpsでも取りこぼしはないようです（Xパラメータ付き）。

このポケコンは一度通信パラメータを設定すると，リセットしない限り内容は保存されているので，あたかもPC−9801(やX68000）のハードディスクをポケコンにつないでいるような感じで使うことができます。

＊

D&D，AD&D以外をやっている人は，がんばってそれ用のプログラムを書いてください。また，このプログラムでは「ヒドラ」はサポートされていません。それから今回のプログラムはPDSとしてあつかってかまいません。ネットにアップするなり（？）改造するなりご自由にどうぞ。

次回はちょっとしたツールや，シナリオの作り方などを書いてみようと思います。

リスト1　マスター戦闘支援ツールCST

```
100 CLS :CLEAR :RANDOMIZE
110 ON ERROR GOTO *ERR
120 PRINT "** combat support tool **"
130 MN=7
140 DIM HP(2,MN),HPORG(2,MN),N(2),STATUS(2,MN),THAC0(2),NAT(2),G
HP(2),GHD$(2),DMGN(2,7),DMGD(2,7),DMGP(2,7),F(MN),R(2)
150 F$="":INPUT "file name/ret=";F$:IF F$<>"" THEN GOSUB *LDATA
160 *LOOP
170 GOSUB *SETDISP
180 *LPMAIN
190 GOSUB *SETLINE:S$="":INPUT "S,A,R,0-2,Q>";S$
200 IF S$="" THEN *LOOP
210 A$=LEFT$ (S$,1):S$=RIGHT$ (S$,LEN S$-1):G=VAL LEFT$ (S$,1)
220 IF A$="S" GOSUB *SET:GOTO *LOOP
230 IF A$="A" GOSUB *ATTACK:GOTO *LOOP
240 IF A$="R" GOSUB *REGENE:GOTO *LOOP
250 IF A$="Q" GOTO *QUIT
260 IF "0"<=A$ AND A$<"9" THEN G=VAL A$:GOSUB *SETDMG:GOTO *LPMA
IN
270 GOTO *LOOP
280 *REGENE
290 IF G<0 OR G>2 RETURN
300 IF S$<>"" THEN R(G)=VAL (RIGHT$ (S$,LEN S$-1)):RETURN
310 FOR G=0 TO 2
320 IF R(G)=0 THEN 350
330 FOR J=0 TO N(G)-1:HP(G,J)=HP(G,J)+R(G):IF HP(G,J)>HPORG(G,J)
 THEN HP(G,J)=HPORG(G,J)
340 NEXT J
350 NEXT G
360 RETURN
370 *SET
380 A$=LEFT$ (S$,1)
390 G=VAL (S$):CPY=0
400 IF G<0 OR G>2 RETURN
410 FOR J=2 TO LEN S$:IF MID$ (S$,J,1)="=" THEN GG=VAL (RIGHT$ (
S$,LEN S$-J)):CPY=1:IF N(GG)=0 THEN RETURN
420 IF N(G) THEN GOSUB *SETLINE:PRINT G;"is now used. OK(Y/N)?":
Y$=INPUT $(1):IF Y$<>"Y" RETURN
430 GOSUB *SETLINE:N=0:INPUT "N=";N:IF N<0 OR N>MN+1 THEN 430
440 N(G)=N:IF N=0 THEN RETURN
450 IF CPY THEN GOSUB *CPY ELSE GOSUB *MSTR
460 RETURN
470 *CPY
480 GHP(G)=GHP(GG):HPG=GHP(GG):GHD$(G)=GHD$(GG):GOSUB *LEVEL
490 NAT(G)=NAT(GG):IF NAT(G)=0 THEN RETURN
500 THAC0(G)=THAC0(GG)
510 FOR J=0 TO NAT(G)-1
520 DMGN(G,J)=DMGN(GG,J)
530 DMGD(G,J)=DMGD(GG,J)
540 DMGP(G,J)=DMGP(GG,J)
550 NEXT J
560 RETURN
570 *MSTR
580 GOSUB *SETLINE:HPG=0:INPUT "HP=(/ret)";HPG
590 GHP(G)=HPG
600 GOSUB *SETLINE:HD$="":PRINT "HD/LV=";:IF HPG>0 THEN PRINT "(
/ret)";
610 HD$="0":INPUT "";HD$
620 IF HPG=0 AND HD$="0" THEN N(G)=0:RETURN
630 GHD$(G)=HD$:GOSUB *LEVEL
640 NAT(G)=0
650 IF HD$="0" THEN 680
```

```
660 GOSUB *SETLINE:INPUT "#at=";NAT(G)
670 IF NAT(G)>0 THEN GOSUB *SETAT ELSE NAT(G)=0
680 RETURN
690 *SETAT
700 FOR J=0 TO NAT(G)-1
710 S$="+000":GOSUB *SETLINE:PRINT "dmg";J+1;"(/N)=";:INPUT "";S
$
720 IF S$="N" AND J>0 THEN J=J-1:GOTO 710
730 IF J=0 OR S$<>"+000" GOSUB *SETNDP
740 DMGN(G,J)=ND:DMGD(G,J)=DT:DMGP(G,J)=P
750 NEXT J
760 GOSUB *SETLINE:T=0:INPUT "TH adj=";T
770 THAC0(G)=THAC0(G)-T
780 RETURN
790 *LEVEL
800 A$=LEFT$ (GHD$(G),1):S$=RIGHT$ (GHD$(G),LEN GHD$(G)-1):IF S$
="" THEN S$="0"
810 DT=8:ND=9999:P=0:CL=0:H$=GHD$(G)
820 IF A$="F" THEN DT=10:ND=9:P=3:CL=1:H$=S$
830 IF A$="C" THEN DT=8:ND=9:P=2:CL=2:H$=S$
840 IF A$="T" THEN DT=6:ND=10:P=2:CL=3:H$=S$
850 IF A$="M" THEN DT=4:ND=10:P=1:CL=4:H$=S$
860 HD=VAL H$
870 IF HD>ND THEN P=P*(HD-ND) ELSE ND=HD:P=0
880 P=P+EVAL H$-VAL H$
890 GOSUB *THAC0
900 THAC0(G)=THAC0
910 FOR J=0 TO N(G)-1
920 STATUS(G,J)=0
930 IF HPG=0 THEN GOSUB *NDP:HPORG(G,J)=R:HP(G,J)=R ELSE HPORG(G
,J)=HPG:HP(G,J)=HPG
940 IF HP(G,J)<=0 THEN HPORG(G,J)=1:HP(G,J)=1
950 NEXT J
960 RETURN
970 *THAC0
980 IF CL>0 THEN *FIG
990 *MON
1000 IF P>=3 THEN HD=HD+1
1010 IF HD>=1 THEN THAC0=19-INT ((HD-1)/2)*2
1020 RETURN
1030 *FIG
1040 IF CL>1 THEN *CLE
1050 THAC0=21-HD
1060 RETURN
1070 *CLE
1080 IF CL>2 THEN *THI
1090 THAC0=20-INT ((HD-1)/3)*2
1100 RETURN
1110 *THI
1120 IF CL>3 THEN *MAG
1130 THAC0=20-INT ((HD-1)/2)
1140 RETURN
1150 *MAG
1160 THAC0=20-INT ((HD-1)/3)
1170 RETURN
1180 *SETNDP
1190 FOR P=1 TO LEN S$
1200 IF MID$ (S$,P,1)="*" THEN *NDPNDP
1210 NEXT P
1220 GOTO *NDPMM
1230 *NDPNDP
1240 ND=VAL S$
1250 FOR P=2 TO LEN S$
1260 IF MID$ (S$,P,1)="*" THEN 1280
1270 NEXT P
1280 S$=RIGHT$ (S$,LEN S$-P)
1290 DT=VAL S$
1300 P=EVAL S$-VAL S$
1310 RETURN
1320 *NDPMM
1330 MIN=VAL S$
1340 IF EVAL S$=MIN THEN MAX=MIN ELSE MAX=-EVAL S$+MIN
1350 ND=0:DT=0:P=0
1360 FOR MM=1 TO 20
1370 P=MIN-MM
1380 DT=(MAX-P)/MM
1390 IF DT=1 OR DT=2 OR DT=3 OR DT=4 OR DT=6 OR DT=8 OR DT=10 OR
 DT=12 OR DT=20 THEN ND=MM:GOTO 1420
1400 DT=0
1410 NEXT MM
1420 IF DT=0 THEN ND=0:DT=0:P=MIN
1430 RETURN
1440 *NDP
1450 R=P
1460 FOR JRND=1 TO ND
1470 R=R+RND DT
1480 NEXT
1490 RETURN
1500 *SETDISP
1510 CLS
1520 FOR G=0 TO 2
1530 LOCATE 0,G:PRINT "GROUP";RIGHT$ (STR$ G,LEN STR$ G-1);
1540 IF N(G)<=0 THEN 1560
1550 GOSUB *DISP
1560 NEXT G
1570 RETURN
1580 *DISP
1590 FOR J=0 TO N(G)-1
1600 LOCATE 7+4*J,G
1610 IF HP(G,J)<=0 PRINT "    ";:GOTO 1630
1620 PRINT HP(G,J);:LOCATE 7+4*J,G:IF STATUS(G,J) THEN  PRINT CH
R$ (97+J); ELSE PRINT CHR$ (65+J);
1630 NEXT J
1640 RETURN
1650 *QUIT
1660 GOSUB *SETLINE:F$="":INPUT "quit:filename/ret/N=";F$
1670 IF F$="" THEN END
1680 IF F$="N" THEN *LOOP
1690 OPEN "E:"+F$ FOR OUTPUT AS #1
1700 FOR G=0 TO 2
1710 IF N(G)<=0 THEN 1790
1720 PRINT #1,GHD$(G):PRINT #1,N(G):PRINT #1,GHP(G):PRINT #1,THA
C0(G):PRINT #1,NAT(G)
1730 FOR J=0 TO NAT(G)-1
1740 PRINT #1,DMGN(G,J);"*";DMGD(G,J);"+";DMGP(G,J)
1750 NEXT J
1760 FOR J=0 TO N(G)-1
1770 PRINT #1,HPORG(G,J)
1780 NEXT J
1790 NEXT G
1800 CLOSE
1810 END
1820 *LDATA
1830 OPEN "E:"+F$ FOR INPUT AS #1
1840 FOR G=0 TO 2
1850 IF EOF (1) THEN CLOSE :RETURN
1860 INPUT #1,GHD$(G):INPUT #1,N(G):INPUT #1,GHP(G):INPUT #1,THA
C0(G):INPUT #1,NAT(G)
1870 FOR J=0 TO NAT(G)-1
1880 INPUT #1,S$:GOSUB *NDPNDP:DMGN(G,J)=ND:DMGD(G,J)=DT:DMGP(G,
J)=P
1890 NEXT J
1900 FOR J=0 TO N(G)-1
1910 INPUT #1,HPORG(G,J):HP(G,J)=HPORG(G,J)
1920 NEXT J
1930 NEXT G
1940 CLOSE :RETURN
1950 *ATTACK
1960 IF G<0 OR G>2 RETURN
1970 IF N(G)=0 RETURN
1980 IF NAT(G)=0 RETURN
1990 FOR J=0 TO N(G)-1
2000 CLS
2010 IF HP(G,J)<=0 OR STATUS(G,J)<>0 THEN *SKPAT
2020 PRINT "G";CHR$ (48+G);" d20 "
2030 PRINT USING "th0###";THAC0(G):PRINT "    dmg"
2040 FOR K=0 TO NAT(G)-1
2050 R=RND 20:IF R=20 THEN T$="!!!" ELSE IF R=1 THEN T$="---" EL
SE T$=STR$ (THAC0(G)-R)
2060 R$=RIGHT$ (" "+STR$ R,2)
2070 LOCATE 7+K*4,0:PRINT "[";R$;"]":LOCATE 7+K*4,1:PRINT " ";T$
2080 ND=DMGN(G,K):DT=DMGD(G,K):P=DMGP(G,K):GOSUB *NDP:LOCATE 6+K
*4,2:PRINT USING "####";R;
2090 NEXT K
2100 IF INKEY$ <>"" THEN 2100:
2110 LOCATE 0,3:PRINT "->";CHR$ (65+J);
2120 LOCATE 10,3:PRINT "hit ret/N!";:A$=INPUT $(1)
2130 IF A$="N" THEN 2150
2140 *SKPAT:NEXT J
2150 RETURN
2160 *SETDMG
2170 IF G<0 OR G>2 RETURN
2180 IF N(G)=0 RETURN
2190 LOOP=0
2200 *LPDMG
2210 IF S$="" THEN GOSUB *SETLINE :PRINT "HPcalc(";G;:INPUT "):"
;S$:LOOP=1
2220 IF S$="" THEN RETURN
2230 FOR J=0 TO N(G)-1 :F(J)=0:NEXT J
2240 A$=LEFT$ (S$,1):IF A$<"A" OR "H"<A$ THEN RETURN
2250 F(ASC A$-65)=1
2260 FOR J=2 TO LEN S$
2270 A$=MID$ (S$,J,1):IF A$<"A" OR "H"<A$ THEN 2300
2280 F(ASC A$-65)=1
2290 NEXT J
2300 IF A$="*" THEN HOLD=1 ELSE HOLD=0
2310 S$=RIGHT$ (S$,LEN S$-J+1)
2320 DMG=VAL S$:SV=21:FIRE=0:D2=INT ((DMG+1)/2)
2330 FOR J=2 TO LEN S$
2340 IF MID$ (S$,J,1)="F" THEN FIRE=1
2350 IF MID$ (S$,J,1)="/" THEN SV=VAL (RIGHT$ (S$,LEN S$-J)):GOT
O 2370
2360 NEXT J
2370 FOR J=0 TO N(G)-1
2380 IF F(J)=0 THEN 2420
2390 D=D2:IF SV>RND 20 THEN D=DMG:IF HOLD THEN STATUS(G,J)=1-STA
TUS(G,J)
2400 HP(G,J)=HP(G,J)-D:IF FIRE THEN HPORG(G,J)=HPORG(G,J)-D
2410 IF HP(G,J)>HPORG(G,J) THEN HP(G,J)=HPORG(G,J)
2420 NEXT J
2430 GOSUB *DISP
2440 IF LOOP THEN S$="":GOTO *LPDMG
2450 RETURN
2460 *SETLINE
2470 LOCATE 0,3:PRINT "                                        ";:LOCATE 0
,3
2480 IF INKEY$ THEN 2480
2490 RETURN
2500 *ERR:PRINT "DISK ERROR!":CLOSE
2510 END
```

D＆Dの場合の変更点

```
820 IF A$="F" THEN DT=8:ND=9:P=2:CL=1:H$=S$
830 IF A$="C" THEN DT=6:ND=9:P=1:CL=2:H$=S$
840 IF A$="T" THEN DT=4:ND=9:P=2:CL=3:H$=S$
850 IF A$="M" THEN DT=4:ND=9:P=1:CL=4:H$=S$

990 *MON
1000 IF P>=3 THEN HD=HD+1
1010 IF HD<=7 THEN THAC0=20-HD ELSE THAC0=16-INT(HD/2)
1020 RETURN
1030 *FIG
1040 IF CL>1 THEN *CLE
1050 THAC0=19-INT((HD-1)/3)*2
1060 RETURN
1070 *CLE
1080 IF CL>2 THEN *THI
1090 THAC0=19-INT ((HD-1)/4)*2
1100 RETURN
1110 *THI
1120 IF CL>3 THEN *MAG
1130 THAC0=19-INT ((HD-1)/4)*2
1140 RETURN
1150 *MAG
1160 THAC0=20-INT ((HD-1)/5)*2
1170 RETURN
```

BACK ISSUES バックナンバー案内

ここには1989年9月号から1990年8月号までをご紹介しました。現在1989年10～12，1990年1～8月号までの在庫がございます。バックナンバーおよび定期購読のお申し込み方法については，174ページを参照してください。

1989

9月号(品切り)

特集　活用ハードディスク&プリンタ
各社ハードディスク接続総チェック/ハードディスク雑学講座/COPYキーメニュー/ビデオプリンタ活用プログラム　他
THE SOFTOUCH　ジェノサイド/琉球/mFORTH Compiler
● サイバースティックで遊ぶ　不思議な環境ソフトの世界
● X1/X1turbo用シューティングゲーム　Defeat X
Z80's Bar/MZ-2500グラフィックエディタ　他
[X68000] X-BASIC/マシン語/C言語講座/DōGA・CGA
全機種共通システム　生物進化シミュレーションBUGS

10月号

特集　ゲーム面白心理学
ソーサリアン・宇宙からの訪問者/ファンタジーゾーンねじ式/ガウディ・バルセロナの風/サバッシュ　他
● MZ-700用シューティングゲームSide Roll-F
● X1/X1turdo用カードゲームBonding
ショートプロ/Z80's Bar/MZ-2500グラフィックエディタ
X68000マシン語/X-BASIC/C言語講座/DōGA・CGA
THE SOFTOUCH　Z'sTRIPHONY DIGITAL CRAFT/James68K
全機種共通システム　小型インタプリタ言語TTI

11月号

特集　microComputer入門
初歩からのCPU物語/RISCプロセッサの設計と製作
X68000&X1で周辺LSIを使いこなそう
連載　ショートプロ/Z80's Bar/MZ-2500グラフィックエディタ
X68000マシン語/X-BASIC/C言語講座/DōGA・CGA
● X68000用カードゲームばばぬき
LIVE in '89　メタルホーク/オブ・ラ・ディ，オブ・ラ・ダ
THE SOFTOUCH　Stationery PRO-68K/リングマスター1
全機種共通システム　TTI用パズルゲームPUSH BON!

12月号

特集　Cプログラミングへの招待
付録　C言語簡易リファレンス
連載　ショートプロぱーてぃ/Z80's Bar
X68000マシン語/X-BASIC/DōGA・CGA
● Oh!X2周年特別企画「素粒子の声が聞こえる」
● X1/turbo用アクションゲームACTIVE UNIT
LIVE in '89　天空の城ラピュタ/ギャラクシーフォース
THE SOFTOUCH　38万キロの虚空/た～みのる2
全機種共通システム　SLANG用リダイレクションライブラリ

1990

1月号

特集1　オペレーティングスタイルの研究
特集2　Cプログラミング応用編
連載　ショートプロぱーてぃ/Z80's Bar
X68000マシン語/C言語講座/DōGA・CGA
● X1/turbo用シミュレーションゲームSuper Battle
LIVE in '90　さよならを過ぎて/RYDEEN
THE SOFTOUCH　レナム/メタルサイト
全機種共通システム　WORM KUN/再掲載SLANG
特別付録　X68000 THE SOFTWARE CATALOGUE

2月号

特集　画像圧縮へのアプローチ
連載　ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
X68000マシン語/C言語講座/X-BASIC調理実習
● X68000用ゲームプログラムGon Gon
● MZ-700用紙芝居Eyelarth
LIVE in '90　オーダイン/魔女の宅急便
THE SOFTOUCH　A-JAX/フラッピー2/夢幻戦士ヴァリスII
マジックパレット/Mu-1/CYBERNOTE PRO-68K
全機種共通システム　超小型コンパイラTTC++

3月号

特集　MUSICアドベンチャー
X68000用MIDIドライバ&音源エディタ
なんでも鳴らせるOPMD.X/MMLを楽譜データに
連載　ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
C調言語講座/X-BASIC調理実習
● X1/turboシミュレーションCRISIS in Tokyo
LIVE in '90　パワードリフト/スキーム/となりのトトロ
THE SOFTOUCH　ナイトアームズ/斬/ダンジョンマスター
全機種共通システム　超多機能アセンブラOHM-Z80

4月号

特集　ゲームシステム文学誌
1989年度GAME OF THE YEAR発表
連載　ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
X-BASIC調理実習/C調言語講座/X68000マシン語
● X1・MZ-2000/2500用RPG The Cave of Dalk
● うわさの68040，ついに登場
LIVE in '90　バーニングフォース(OPMD対応)
THE SOFTOUCH　The Fille Professor/HOST PRO-68K
全機種共通システム　ファジィコンピュータシミュレータI-MY

5月号

特集　BASICプログラミング
第5回　言わせてくれなくちゃだワ
連載　ショートプロぱーてぃ/Z80's Bar
X-BASIC調理実習/X68000マシン語プログラミング
● 新機種X68000SUPER-HD/EXPERTII/PROII
● ラジコンスティックの製作
LIVE in '90　TURBO OUTRUN
THE SOFTOUCH　天下統一/ポピュラス/Hyperword
全機種共通システム　インタプリタ言語STACK

6月号

特集　創刊8周年記念PRO-68K(付録5"2HD)
Oh!Xアンケート結果大分析大会
連載　ショートプロぱーてぃ/Z80's Bar/PurePASCAL
X-BASIC調理実習/X68000マシン語プログラミング
● X1turbo用コマンドシェルシミュレータ
● ハードウェア工作入門
LIVE in '90　ナイトアームズ/悪魔城伝説/この木なんの木
THE SOFTOUCH　三国志II/FAR SIDE MOON/グラナダ
全機種共通システム　X68000用S-OS"SWORD"他

7月号

特集　マシン語への第一歩
X68000SUPER-HD試用レポート
連載　ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
X-BASIC調理実習/PurePASCAL
● INTEGRAL X1――ノーマルX1への対応
● ハードウェア工作入門
LIVE in '90　夢幻戦士ヴァリスII/トッカータとフーガニ短調
THE SOFTOUCH　サーク/あーくしゅ/ダウンタウン熱血物語
全機種共通システム　リロケータブルアセンブラWZD

8月号

特集　ADVANCED 2D GRAPHICS
100号記念特別モニタプレゼント
連載　ショートプロぱーてぃ/Z80's Bar/INTEGRAL X1
X-BASIC調理実習/X68000マシン語プログラミング
PurePASCAL/ハードウェア工作入門
● X68000用画像回転プログラム　XROT0.X
LIVE in '90　OMENS OF LOVE/ENDLESS RAIN/ダートフォックス
THE SOFTOUCH　大航海時代/ウルティマV/プロミストランド
全機種共通システム　リンカWLK

郵便はがき

料金受取人払

高輪局承認

1459

差出有効期間
平成4年7月
15日まで

108-00

(受取人)

東京都港区高輪
2-19-13 NS高輪ビル

ソフトバンク株式会社

電話

住所

氏名　　　　年齢

職業・勤務先
学校・学部・学年

●編集部へのメッセージ

今月号の特集について

いちばん良かった記事	興味のなかった記事
これから載せてほしい記事内容	本誌以外にお読みのパソコン雑誌

推薦する市販ソフト

ソフト名：

推薦理由：

初めて買ったパソコンは何ですか？

あなたの愛機は(所有機種に○印をつけてください)　　ない
X1(マニアタイプ,C,D,F,G,twin) X1turbo(model 10,20,30,40,II,III,Z,ZII,ZIII)
MZ-(80K/C, 1200, 700, 1500, 80B, 2000, 2200, 2500, 2861)
X68000(初代,ACE,PRO,PROII,EXPERT,EXPERTII,SUPER,HD)　その他
FD(　基)　TAPE　QD　HD(　MB)　プリンタ(　)

年齢　歳　パソコン歴　年　男・女　プレゼントNo.

振替用紙

←点線から、きれいに切り取ってご使用ねがいます。

払込票

通常払込料金
加入者負担

口座番号	東京 1－29307
加入者名	ソフトバンク株式会社
金額	億 千 百 十 万 千 百 十 円
払込人住所氏名	※
備考	受付局日付印

切り取らないで郵便局にお出しください。

記載事項を訂正した場合は、その箇所に訂正印を押してください。

払込通知票

通常払込料金
加入者負担

口座番号	東京 1－29307	金額	億 千 百 十 万 千 百 十 円
加入者名	ソフトバンク株式会社	料金	払込み / 特殊 円
払込人住所氏名	※ （郵便番号 ）	備考	
		受付局日付印	

各票の※印欄は、払込人において記載してください。

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください また、本票を折り曲げたりしないでください。 （郵政省）

切り取り線

定期購読のお申し込みを頂きありがとうございます。この申込書の弊社到着〆切は次の通りです。これを過ぎますと次号からの発送となりますので、ご了承下さい。尚、郵便振替は郵便局で払い込まれてから当社に到着するのに、2週間位かかります。

Oh! PC	1日発売	前月 20日
	15日発売	当月 5日
月刊情報処理試験 / BEEP メガドライブ		前月 末日
Oh! X / Oh! FM / THE COMPUTER / C MAGAZINE / パソコン マガジン		当月 10日

(御注意)

当月発売の号より前に遡ってからの申し込みは出来ません。

バックナンバーは、お近くの書店でお申し込み下さい。

切り取らないで郵便局にお出しください。

この欄は、加入者あての通信にお使いください。

送り先				
お名前	フリガナ	性別 男・女	年齢	ご職業
ご住所	フリガナ 〒			
お電話	ご自宅	勤務先		

申込書			
THE COMPUTER 定期購読	新規申し込み	継続申し込み TC NO.	年間 7,200円
Oh!PC 定期購読	新規申し込み	継続申し込み PC NO.	年間11,440円 6ヶ月 5,720円
Oh!X 定期購読	新規申し込み	継続申し込み X NO.	年間 6,720円
Oh!FM 定期購読	新規申し込み	継続申し込み FM NO.	年間 6,720円
月刊情報処理試験 定期購読	新規申し込み	継続申し込み JS NO.	年間 8,160円 6ヶ月 4,080円
C MAGAZINE 定期購読	新規申し込み	継続申し込み CM NO.	年間11,760円
パソコンマガジン 定期購読	新規申し込み	継続申し込み PM NO.	年間 6,960円
Beep メガドライブ 定期購読	新規申し込み		年間 5,760円
通信欄			

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。　（郵政省）

愛読者プレゼント

プレゼントの応募方法

とじ込みのアンケートはがきの該当項目をすべてご記入のうえ，希望するプレゼント番号をはがき右下のスペースにひとつ記入してお申し込みください。締め切りは1990年9月18日の到着分までとします。当選者の発表は1990年11月号で行います。

❶ ギャラガ'88

電波新聞社　☎03(445)6111

X68000用 5"2HD版2枚組

8,200円

3名

ナムコの大ヒットシューティングゲームのリメイク版。X68000用にアレンジされたダンシングステージも見ものだぞ。

❷ ウルティマV

ポニーキャニオン　☎03(221)3161

X68000用 5"2HD版2枚組

9,800円

3名

根強い人気のあるウルティマシリーズ第5作。これはウルティマⅣの続編となっている。もちろんⅣのデータが利用できるぞ。

❸ D-Again

データウエスト　☎06(968)1236

X68000用 5"2HD版4枚組

8,800円

3名

いよいよ第4のユニットシリーズもこのD-Againをもって第1部WWWF編完結。ブロンウィンファン必携のアドベンチャーだ。

❹ システムソフト スーパーシュミレーション・サウンド

システムソフト　☎092(714)6236

2,718円

5名

大戦略Ⅱ，大戦略Ⅲ，そして天下統一のゲームミュージックを収めたCD。発売元は株式会社メルダック。

❺ 虹色ディップスイッチ

ビジネス・アスキー　☎03(486)7119

971円

5名

あのドラクエで有名な堀井雄二が，ゲーム業界の裏話を書いたエッセイ集。ゲーム業界に興味のある人でなくても，十分楽しめる本だ。

7月号プレゼント当選者

❶ダウンタウン熱血物語（愛知県）行松徹（和歌山県）木村誠吾（広島県）山川剛信　❷サーク（北海道）山崎康則（大阪府）岩田薫（長崎県）帆足慎二　❸天下統一（東京都）小島尚基（神奈川県）赤城豊和（徳島県）谷口成広　❹ファルコムシステム手帳（宮城県）真野勝昭（岐阜県）藤掛弘隆　❺メモ帳（福島県）根津正博（栃木県）毛塚健次（石川県）中村学（山口県）藤沢邦昭（熊本県）松本英貴

以上の方々が当選されました。おめでとうございます。商品は順次発送いたしますが，入荷状況などにより遅れる場合もあります。また，公正取引委員会の告示により，このプレゼントに当選された方は，この号の他の懸賞には当選できない場合がありますのでご了承ください。

（価格はすべて消費税別です）

PENGUIN INFORMATION CORNER
ペ・ン・ギ・ン・情・報・コ・ー・ナ・ー

NEW PRODUCTS

64KバイトRAM標準装備
PC-E550
シャープ

PC-E550

シャープはすでに発売中の高機能関数ポケットコンピュータ「PC-E500」の新機種として「PC-E550」を発売した。

「PC-E550」は64KバイトRAMを標準装備。さらに，RAMカードを装着することで最大128Kバイトまで拡張可能となっている。もちろん，PC-E500用のソフトはそのまま使用可能である。また，外観はホワイトデザインが採用されている。価格は32,000円（税別）。

〈問い合わせ先〉
シャープ㈱　☎03(260)1161,06(621)1221

ハンディターミナル
HC-70
セイコーエプソン

セイコーエプソンは16ビットCPUのV25を搭載し128×192ドットの大画面STN液晶パネルを採用したハンディターミナル「HC-70」を発売した。

「HC-70」では256Kバイトの漢字ROMを搭載しJIS第1水準はもとより，JIS第2水準漢字もサポートしている。メモリは本体内蔵RAM256Kバイト，拡張メモリ用オプションとしてRAMカード128Kバイトおよび256Kバイトが用意されている。用途に合わせてRAMカードを使用することにより，データの集積，管理が容易に行える。さらに，RS-232Cインタフェイス，メモリカードインタフェイス，光通信インタフェイスを標準装備。入力データの転送速度の違いによって通信方法を簡単に選択することができる。

この「HC-70」のソフト開発はパソコン上で行うことができる。対応機種は「エプソン PC-386/286シリーズ」，「NEC PC-9801シリーズ」。また，開発支援用ツールとして「HC-70専用ライブラリ」が用意されている。

オプションは以下のものが用意されている。

・プリンタユニット
・光通信ユニット
・RAMカード128Kバイト/256Kバイト
・充電器
・通信ケーブル
・キャリングケース，ストラップバンド

外形寸法203.5×86.0×27.5mm，重量は本体約410g，電池パック約120g。連続使用時間（電池寿命）約15時間。価格は220,000円（税別）。

〈問い合わせ先〉
セイコーエプソン㈱　☎0266(58)1705
エプソン販売㈱　☎03(377)7001

HC-70

ノートワープロ
WV-700
シャープ

WV-700

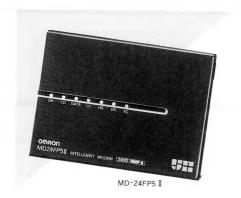
シャープはノートワープロ「WV-700」を9月1日に発売する。本体とプリンタ・フロッピーディスクドライブユニットが必要に応じて分離可能，また厚さ30mm，重さ1.4kgというコンパクトサイズの実現により，携帯性を向上している。電源は単三アルカリ乾電池,充電式バッテリーパック（別売），ACアダプタが使用可能で，最高連続使用時間はバックライトオフ時で約8時間（オン時は4時間）を実現している。価格は218,000円（税別）。

〈問い合わせ先〉
シャープ㈱　☎03(260)1161,06(621)1221

ポケットモデム
MD24FP4II/5II
オムロン

オムロンはポケットモデム「MD24FP4」のバージョンアップシリーズとして「MD24FP4II/5II」を発売した。いずれの機種もエラー訂正機能としてMNPと最新の国際標準規格CCITT V.42を搭載している。特に「MD24FP5II」はMNPクラス5を標準搭載し，その圧縮機能により実効通信速度が4800bpsまで向上させ，信頼度も高めている。

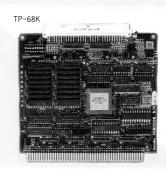
さらに，NCUはAA（自動発着信）だけでなくMA（手動発信・自動着信），MM（手

MD-24FP5 II

動発着信)にも対応している。価格は38,800円と42,800円（いずれも税別）。
〈問い合わせ先〉
オムロン㈱　☎03(5488)3221

電子手帳用ICカード
PA-3C19/22〜24/26〜29
シャープ

電子システム手帳用ICカード

電子システム手帳用ICカードの第３弾としてソフト開発会社７社より８機種が発売される。
・ビジネス英会話カード
英会話約2,500文例，英単語約1,300語，世界の都市情報を収録。
㈱学習研究社
標準価格9,000円(税別) 発売中
・「パズル大迷宮」カード
全100面,エジプトの砂漠を舞台にしたアドベンチャー仕立てのパズル。
㈱アスク講談社
標準価格6,000円(税別) 発売中
・電子易占いカード「易道」
中国５千年の歴史を持つ易学を誰にでも簡単に。
㈱アスミック
標準価格9,000円(税別) ８月発売
・旺文社の入試に出る古文単語300
旺文社刊行の「入試に出る古文単語300」(中村幸弘著) をICカード化。
ソフトバンク㈱
標準価格8,500円（税別）　９月発売
・株式カード「キャピタルゲイン」
株価のグラフ，利益管理のための計算，さらには税や手数料の計算などを簡単に。
日本データ機器㈱
標準価格8,500円（税別）　９月発売
・上海IIカード
皆さんお馴染みの「上海」がパワーアップした「上海II」がICカードに。
サン電子㈱
標準価格7,000円（税別）　８月発売
・TALKING BUSINESSカード
英会話約1,600文例，英単熟語約4,400語(うち700語は使用されている文例を検索可能)を収録したビジネス英会話カード。
㈱アスミック
標準価格9,000円（税別）　10月発売
・多機能メモカード「ハイパーメモ」
PA-8600の約２倍の記憶容量を持ち，アイデア次第でさまざまな情報を記憶，活用できる多機能メモリカード。
㈱マイクロキャビン
標準価格16,000円（税別）　10月発売
〈問い合わせ先〉
シャープ㈱　☎03(260)1161,06(621)1221

X68000用アドインボード
TP-68K/K-TRAM-2
国際データシステム

TP-68K
K-TRAM-2
国際データシステムはX68000対応の大容量メモリ付きトランスピュータアドインボード，拡張モジュール，およびパラレルCコンパイラを８月より発売する。

アドインボードMODEL　TP-68Kは4MDRAMを採用することによりオンボードに基本4Mバイト，最大8Mバイトの大容量メモリを搭載した。トランスピュータの高速処理性と大容量ローカルメモリによりX68000によるレイトレーシングなどを高速化する。価格は428,000円より。

拡張モジュールMODEL K-TRAM-2はアドインボードに装着することによりマルチCPU化（並列化）するサブボードである。パラレルプログラミングの容易なコンピュータグラフィック関係ではトランスピュータのマルチ数にほぼ比例してスピード向上が得られる。価格は260,000円より。

パラレルCコンパイラはトランスピュータ用パラレルコンパイラとして普及している英国3L社製コンパイラを移植し，XCのグラフィック関係コマンドを取り込んだもの。Cコンパイラとしては K&Rに完全準拠している(ANSIではない)。価格は268,000円（すべて税別)。
〈問い合わせ先〉
国際データシステム㈱　☎0423(76)4800

INFORMATION

第2回　サイクロンCG大会

アンス・コンサルタンツが主催する「サイクロンCG大会」が今年も開催される。開催の日時は９月24日(月)午後１時半より６時まで，場所は昨年と同じ東京渋谷道玄坂のフォーラム８（新大宗ビル）となっている。

サイクロンCG大会はレイトレーシングツール「サイクロン」のユーザーを対象にしたCG作品コンテストで，昨年の第１回大会ではX68000版のサイクロンユーザーを中心として質の高い作品に恵まれ盛り上がりを見せた。

今回は，PCおよびFMシリーズのユーザーも含め，よりスケールアップした大会となる予定。賞金20万円のグランプリほか賞金・賞品も多数用意されている。

応募作品の締め切りは９月８日で，以下の宛先まで郵送のこと。
〒810 福岡市中央区平丘町68
アンス・コンサルタンツ

FILES Oh!X

このインデックスは，タイトル，注記——筆者名，誌名，月号，ページで構成されています。受験生の方，夏休みの間思うように勉強ははかどりましたか？　あと半年の辛抱です。頑張ってください。

参考文献
I/O　工学社
ASCII　アスキー
コンプティーク　角川書店
The BASIC　技術評論社
テクノポリス　徳間書店
POPCOM 7月号　小学館
マイコン　電波新聞社
マイコンBASIC Magazine　電波新聞社
LOGIN　アスキー

一般

▶ネットワーカー・ホリック第23回
PDS特集第2弾。LHARCアーカイバ。中山美穂のラジオ番組と連動したネット「中山美穂　P.S. I LOVE YOU」を紹介。——編集部，LOGIN，13号，204-205pp.

▶特集　記憶するものたち
フロッピー，CD-ROM，ハードディスク，ICカード，光磁気ディスクといった，コンピュータに不可欠な記憶メディアの構造，製造法などを詳しく解説。——編集部，LOGIN，14号，128-141pp.

▶ネットワーカー・ホリック第24回
大手ネットのフォーラム最新情報。NIFTY-Serveのぴあ関連コーナーやプロレスフォーラム，Rupo-NETなど。PDSはX68000の3Dポリゴンフライトシミュレータ「CAPTAIN」。全国BBS探訪記は東京麹町の「SC-NET」を紹介。——編集部，LOGIN，14号，216-219pp.

▶My Opinion日コン連にもの申す
日コン連の実情に対する日コン連反主流派(?)からの内部告発記事。——Mr.X編集委員，The BASIC，8月号，172-174pp.

▶NEW PRODUCTS
はがき，リフィルへの印字が可能な電子手帳用プリンタ，シャープCE-80Pを紹介。——編集部，マイコンBASIC Magazine，8月号，96p.

▶イメージスキャナの徹底研究
いまや画像処理に必須になったイメージスキャナの仕組み，インタフェイスの問題についての解説記事。製品と画像サンプルの紹介も行う。——山田憲一，マイコン，8月号，113-128pp.

▶ビジネスツールレポート
電子手帳の新潮流，キヤノンAiノートの機能，文字入力の実力，実用性について考察する。——佐田守弘，マイコン，8月号，151-158pp.

▶実戦ハード入門
光センサを使って風力・風向計を作る。PC-E200などでも測れるぞ。——石川至知，マイコン，8月号，321-325pp.

▶MICOM WATCHING
NECの新本社“NECスーパータワー”のユニークなデザインや，ネットワークサービスの秘密に迫る。——菊池秀一，マイコン，8月号，316-319pp.

▶ノート型パソコンの正しい使い方
急速に普及しつつあるノートパソコン。スペックの違いやそれぞれの特長に合った使い方を研究する。——編集部，ASCII，8月号，257-279pp.

▶どーするどーする研究所
わからなくならないフロッピーの管理法は？　ディスクはどうやって運ぶのがいいか？　フロッピーはどれくらい丈夫か？　などの疑問を解き明かす。——一文字真，ASCII，8月号，289-296pp.

▶PRODUCTS SHOWCASE
シャープAll in Noteのハードウェアレポート。非常に革新的なスタンスに評価が高い。——編集部，ASCII，8月号，301-304pp.

▶PRODUCTS SHOWCASE
最新光磁気ディスク事情と称して，8機種のMOドライブの機能と動向を探る。——編集部，ASCII，8月号，305-307pp.

▶AV STRASSE
音声多重ディスプレイCZ-613D-TNとOS9-VSEDの紹介と，GUNSHIPのX68000版登場のニュースなど掲載。——編集部，ASCII，8月号，375-380pp.

MZシリーズ

MZ-1500（MZ-5Z001 BASIC）

▶UP-DOWN
ダイヤをすべて取って始めの位置に戻る。ショートプログラムゲーム。——川辺悟，マイコンBASIC Magazine，8月号，128p.

MZ-2500（M25-BASIC）

▶しらいむの野望
全国をひとつにまとめるべし。戦国シミュレーションゲーム。——白井俊明，マイコンBASIC Magazine，8月号，129-131pp.

X1/turbo/Z

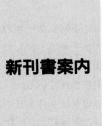
X1シリーズ

▶HOLE ON WALL
7種類のブロックを使って壁にあいた穴をふさぐ。風変わりなパズルゲーム。——JIRONKA，マイコンBASIC Magazine，8月号，155-157pp.

▶マグネッタ君
リモコンロボットを操作して資源をすべて取る。磁力パズルゲーム。——BANCO，マイコンBASIC Magazine，8月号，158-160pp.

▶誌上公開質問状
CU-14ED，CU-14FDはX68000と接続できるか？　X1Gモデル30で機械語モニタからBASICに戻らずにデータをセーブすることができるか？　などに答える。——多田太郎，マイコンBASIC Magazine，8月号，301p.

X1＋FM音源ボード（要NEW FM音源ドライバ）

▶マーベルランド　～アスレチック面～
ナムコのアクションゲーム，ミュージックプログラム。——小室和生，マイコンBASIC Magazine，8月号，184

ロココ町は5年前廃園になったロココランド跡地にできた都市。「超遊園地都市」。東京は遊園地都市の一形態で，ロココ町は「東京の可能世界」のひとつ。「都市は境界のない拡大された遊園地」。作者が東京ディズニーランドを念頭においていることは間違いない。ディズニーランドは都市としては安全で人畜無害すぎる。人々は住人を演じているだけでディズニーランドは人々のある部分を解放する代わりに，ほかの部分を抑圧する。ロココ町はそうではない。常識を越えて住民を教育し，解放する。思考回路や行動形態を解放して生まれ変わる。そして，ロココ町の一部にならなければならない。それをスムーズに行うために「遺伝子分析」なるものがある。遺伝子分析によってあるべき自分をみつけ，ありのまま快楽するなりなんなりする。

センスのない時代遅れのコピーライターのために帯に“SF”と書いてあるが，これはSFではない。確かにひと昔前ならSFとしか分けられなかったが，今は一般文学として成り立つ時代になったのだ。そして，これは今を捉えるに欠かせない都市と遊園地を融合した現代文学のひとつである。　(K)

ロココ町　島田雅彦著　集英社　☎03(230)6100
四六判　243ページ　1,200円

-185pp.

X1turboシリーズ

▶最新ゲーム徹底解剖!!
大航海時代，セレクテッドソーサリアン4を徹底解剖。——編集部，LOGIN，13号，184-185，198-199pp.

▶How To Win
セレクテッドソーサリアン4，三國志IIの紹介。——編集部，コンプティーク，8月号，138-141，150-152pp.

▶ボパの塔
色のついたパネルをすべて青に変える，パズルゲーム。——中西弘幸，マイコンBASIC Magazine，8月号，161-162pp.

X68000

▶NEW SOFT
ポピュラスデータ集「ポピュラス・プロミストランド」とズームの新作ロールプレイング「ラグーン」を紹介。——編集部，LOGIN，13号，17-18pp.

▶X68000新聞
北海道にあるソフトメーカー「ズーム」と「デービーソフト」を訪ねる。そのほかポピュラス・プロミストランド，トンネルズ＆トロールズ，ギャラガ'88，Vessel，GUNSHIP，を紹介。——編集部，LOGIN，13号，130-135pp.

▶最新ゲーム徹底解剖!!
トンネルズ＆トロールズを徹底解剖！——編集部，LOGIN，13号，190-191pp.

▶NEW SOFT
新着ゲームの紹介。ギャラガ'88，ウルティマV，9月発売のレインフォーサー。——編集部，LOGIN，14号，20-23pp.

▶X68000新聞
シムシティーX68000版の発売発表。新着ゲームのサイバリオン，プロテニスワールドコート，ジェミニウイング，闇の血族。——編集部，LOGIN，14号，146-151pp.

▶最新ゲーム徹底解剖!!
トンネルズ＆トロールズの攻略法レポート。——編集部，LOGIN，14号，196-199pp.

▶先取りおすすめゲーム
ポピュラスのシナリオデータ集「ポピュラス・プロミストランド」を紹介。——編集部，テクノポリス，8月号，8-10pp.

▶GAMING WORLD
新着ゲームのギャラガ'88，プロテニスワールドコートを紹介。——編集部，テクノポリス，8月号，20-21pp.

▶新作ゲーム先取り! Soft Flash
制作中のソフトを紹介。銀河英雄伝説II，スペースローグ，クォータースタッフ，パイピヤン，サイバリオン，Thrice，イメージファイト。——編集部，テクノポリス，8月号，27-29pp.

▶攻略おすすめゲーム
ウルティマVを攻略。——編集部，テクノポリス，8月号，54-56pp.

▶How To Win
ポピュラス・プロミストランドの紹介。——編集部，コンプティーク，8月号，146-148pp.

▶X68000 SPIRITS
新着ゲーム，サイバリオン，プロテニスワールドコート，天下統一，エメラルド・ドラゴンを紹介。——編集部，コンプティーク，8月号，248-249pp.

▶WE ARE THE X68000 WORLD
ギャラガ'88，プロテニスワールドコート，闇の血族・上巻，GUNSHIPと8月発売のサイバリオン，Thrice。——編集部，POPCOM，8月号，90-93pp.

▶ミュージック・パビリオン
ミュージックプログラム。「パパ」By PRINCESS²。——編集部，POPCOM，8月号，147-150pp.

▶SPACE WISP
単純な敵よけワンキーゲーム。——菅野類，マイコンBASIC Magazine，8月号，163-164pp.

▶Legend of Adventure
旅の途中に立ちふさがる騎士を倒して王女を救い出す。ジョイスティック専用。アクションゲーム。——林純一，マイコンBASIC Magazine，8月号，165-167pp.

▶恋のホットロック
コナミのゲームミュージックプログラム。——中西道一，マイコンBASIC Magazine，8月号，186-188pp.

▶誌上公開質問状
X68000SUPER-HDはBIOSの改良が行われたが，これによりゲームのスピードも高速化されるか? X68000とX1twinの両方が接続できるディスプレイは？ などの質問に答えている。——多田太郎，マイコンBASIC Magazine，8月号，300-301pp.

▶ギャラガ'88
電波新聞社から発売されたシューティングゲーム「ギャラガ'88」のメインプログラマ佐伯氏のインタビュー。——MUNEPI，マイコン，8月号，186-187pp.

▶X68000マシン語入門
グラフィック加工プログラム集。階調ダウン，色平均化，画面とファイルの重ね合わせプログラムが収録されている。——高橋雄一，マイコン，8月号，290-300pp.

▶なんでもQ&A
ビデオボードとカラーイメージユニットの違いは？ 現在のパレットコードを調べたいのだが？ などの質問に答える。——編集部，マイコン，8月号，386-387pp.

▶Human68k VS MS-DOS
Human68kについて，MS-DOSとの相違点・類似点を検証，Humanならではのバッチファイルの使用法などを紹介する。——吉沢敏男，I/O，8月号，97-110pp.

▶HDC
高速ディスクコピープログラム。必要ない動作を省けるので効率的。なお，セクタをずらした変則フォーマットもできる。——市原昌文，I/O，8月号，161-165pp.

▶Graphメーカー
データを入力すると折れ線グラフを描画してくれるという理工系学生の必需品，グラフ作成ソフト。——西方茂樹，I/O，8月号，153-157pp.

▶コンピュータ・ビールス対策
最近世間を賑わせているコンピュータウイルスの動作・対策・退治法とIPL&SRAMチェックプログラムの掲載。——市原昌文，I/O，8月号，193-208pp.

▶PUSHD POPD
マーキング方式のディレクトリ移動プログラム。面倒なコマンド入力がないのが魅力。——井本裕司，I/O，8月号，248-252pp.

▶俺ってドラムスコ
ジョイパッドをたたいてFM音源のドラムを鳴らそう。——宮城良行，I/O，8月号，256p.

▶ENV.X
環境変数を，あたかもBASICの文字列のように簡単に操作するためのツール。——高橋美幸，I/O，8月号，266-268pp.

▶G98SX
SX-WINDOW用のグラフィックローダ。PC-9801のBRGデータを".PIX"形式に変換する。——Zenobia，I/O，8月号，270-271pp.

▶LOAD TEST
X68000の長期試用レポート。第1回の今回は購入してから，同梱のマニュアルのチェック，SX-WINDOWの試用まで。——編集部，ASCII，8月号，426-429pp.

ポケコン

PC-E500

▶ガンバレ セイビイン
荷物と自分を，入れ替えたりしながら，指定どおりに並び換えるパズルゲーム。——町野稔，マイコンBASIC Magazine，8月号，169p.

▶AIR-WOLF
ヘリコプターで敵をぶっつぶせ！ バリバリシューティングアクションゲーム。——海上貴信，マイコンBASIC Magazine，8月号，170-171pp.

▶表計算プログラムの活用（4）
5月号のPC-E500表計算プログラムとPC-9801のアシストカルクを使ってデータ交換をやってみる。——塚田洋一，マイコン，8月号，327-331pp.

数字オンチの諸君！

「数字オンチとは，数や確率といった基本的な概念をうまく扱えないことをいう」。こういう人は算数の教育レベルが高い（笑）らしい日本でも実に多い。よって，いい加減な数字やエセ科学にみんなコロリと騙される。本書はそういった人々に冷たく優しく数字の概念を身につける大切さとそれがなんら難しくないことを説く。ひとつひとつ挙げるとキリがないが，理科系人間の良い点が随所に発揮された良書である。みんな必読！ (K)

ジョン・アレン・パウロス著 野本陽代訳 草思社 ☎03(470)6565 四六判 190ページ 1,400円

パソコン少年のコスモロジー

コンピュータを代表とするハイテクの持つ可能性に魅了される人は多い。彼らはコンピュータによって変わる精神や生活に視点を注ぐ。本書もそう。パソコン少年に関する考察はイマイチだが，ポイントは後半。バーチャルリアリティなど「パソコンを通して，人類のコスモロジーに転換がおこること」を示そうとする。秀逸なのが「めまい」の概念。情報化は従来のさまざまな概念に「めまい」を起こすという。そのとおりだと思う。 (K)

奥野卓司著 筑摩書房 ☎03(5687)2670 四六判 202ページ 1,440円

QUESTION and ANSWER

X68000で常駐プログラムを作ろうと思ったのですが，作り方がまったくわからないので困っています。できることならサンプルプログラムも交えて，常駐プログラムの作り方を説明してください。　東京都　須藤　和也

普通にプログラムを終了させる場合はDOSコール$FF00(_EXIT)を使いますね。このDOSコールはプログラムの確保していたメモリ領域をすべて開放してしまいますから，プログラムを常駐させることなどできはしません。常駐させたいときはDOSコール$FF31(_KEEPPR)をDOSコール$FF00の代わりに使います。

使い方は以下のようになっています。

```
move.w  code,-(sp)
move.l  prglen,-(sp)
dc.w    $FF31
```

codeにはプログラムの終了コードを指定します。正常終了のときは0を指定しておきましょう。prglenにはプログラムを常駐させる範囲を指定します。引数を必要とするDOSコールを使ったあとはスタックの補正をしなくてはいけませんが，このDOSコールでは必要ありません。

常駐プログラムのサンプルがリスト1です。エディタを使って入力，アセンブル，リンクして実行可能ファイルを作ります。実行の前にCTRL＋F7を2回押して，画面に出ているファンクションキー表示を消します（こうしないと実行結果がわかりづらいからです）。実行するとHuman68……，と表示されるでしょう。文字の色がしだいに青みを増していき真っ青になったかと思うと，今度は逆に淡くなってついには見えなくなってしまいます。するとまたもや青みを増してくる。この動作を延々と繰り返すだけのプログラムです。

プログラムが常駐しているのを確認するには，システムディスク（Ver.2.0以降）のBINディレクトリにあるPROCESSを実行します。常駐しているプログラムはKEEPと表示されます。

このプログラムは常駐部分と非常駐部分で構成されています。常駐部分ではパレットの変更をしています。非常駐部分ではHuman……を表示し，垂直同期割り込みの設定，プログラムの常駐処理（ラベルskip以下5行です）をします。

ここでは常駐範囲をinit−startを計算して求めていますが，私だってもっと簡単に，

```
move.l #init-start,-(sp)
```

と書きたいのです。しかし，初期型同梱のAS.Xでは誤動作します。この書き方では直後にソースリストにないコードが2ワード（イミディエイトデータの値がそっくりそのまま）埋め込まれてしまうのです。

そのほかにプログラムを常駐させるときに注意しておきたいことをいくつか挙げておきましょう。

いままで特に触れませんでしたが，DOSコール$FF31で常駐することのできる範囲は，プログラムの先頭から引数prglenバイト分です。そういうわけで，プログラムを組むときはまず常駐させる部分から書き始めるのが私のやり方です。しかし，実行は非常駐部分のinitからしなくてはいけません。それには最終行の.endに続けてinitを指定します。これで実行アドレスがinitになります。

プログラムを組む場合ほとんどの人がメイン処理やサブルーチン群を前半に，ワークやデータを後半に置いていくと思いますが，常駐プログラムで使うワークやデータ

```
==================  test1.s  ==================
     1: * sample1.s
     2:
     3: _EXIT:          equ     $ff00
     4: _KEEPPR:        equ     $ff31
     5: _B_PRINT:       equ     $21
     6: _VDISPST:       equ     $6c
     7: _G_CLR_ON:      equ     $90
     8: _GPALET:        equ     $94
     9: _SYMBOL:        equ     $bd
    10:
    11:         .text
    12:         .even
    13:
    14: *************************************
    15:
    16: start:  movem.l d0-d2/a1,-(sp)
    17:         move.w  #1,d1
    18:         move.w  palet,d2
    19:         tst.w   flag
    20:         bne     pal2
    21:         addq.w  #2,d2
    22:         cmpi.w  #63,d2
    23:         ble     pal3
    24: pal1:   move.w  #1,flag
    25: pal2:   subq.w  #2,d2
    26:         cmpi.w  #1,d2
    27:         bne     pal3
    28:         clr.w   flag
    29: pal3:   andi.w  #63,d2
    30:         move.w  d2,palet
    31:         move.l  #_GPALET,d0
    32:         trap    #15
    33:         movem.l (sp)+,d0-d2/a1
    34:         rte
    35:
    36: palet:  dc.w    1
    37: flag:   dc.w    0
    38:
    39: * ここまでが常駐部分
    40: *************************************
    41:
    42: *************************************
    43: * ここからが非常駐部分
    44:
    45: init:   moveq   #_G_CLR_ON,d0   * グラフィック画面を
    46:         trap    #15             *  クリアして表示する
    47:         lea.l   symdata,a1      * X68000
    48:         moveq   #_SYMBOL,d0     *  を表示する
    49:         trap    #15             *
    50:         tst.l   d0
    51:         bne     error           * グラフィックが使えない
    52:
    53:         lea.l   start,a1
    54:         move.w  #1*256+2,d1
    55:         move.l  #_VDISPST,d0
    56:         trap    #15             * 割り込み設定
    57:         tst.l   d0
    58:         beq     skip
    59:         lea.l   mes1,a1         * 割り込みが使用中
    60:         move.l  #_B_PRINT,d0    *
    61:         trap    #15             * メッセージを表示
    62:         dc.w    _EXIT           * 常駐せずに終了
    63:
    64: skip:   clr.w   -(sp)           * startからinitまで
    65:         move.l  #init,d1        *  を常駐させて
    66:         sub.l   #start,d1       *  終了する
    67:         move.l  d1,-(sp)
    68:         dc.w    _KEEPPR
    69:
    70: error:  lea.l   mes2,a1
    71:         moveq   #_B_PRINT,d0
    72:         trap    #15
    73:         dc.w    _EXIT
    74:
    75: mes1:   dc.b    '垂直同期割り込みは既に使用されています.
',13,10,0
    76:
    77: mes2:   dc.b    'グラフィックは使用不可能です.',13,10,0
    78:
    79: symdata:dc.w    0,16*31
    80:         dc.l    x68
    81:         dc.b    2,1
    82:         dc.w    1
    83:         dc.b    1,0
    84:
    85: x68:    dc.b    'Human68k V2.01',0
    86:
    87:         .end    init
```

は常駐プログラムに続くように置いてください（サンプルではpalet，flagが常駐するワーク）。こうしないとワークやデータの常駐ができなくなってしまいます。

同じファイルが何個も常駐できてしまうこともありがちです。それを防ぐには実行するファイル名と同じファイル名のプログラムが，現在メモリ上にあるか調べればいいのです。それにはメモリ管理ポインタを遡ってファイルを検索照合すればいいのですが，たいてい6，7文字のファイル名がつけられていることもあって，照合するにはちょいと面倒です。そこで考えられたのがプログラムの先頭にヘッダをつけて，それを検索照合していく方法です。ヘッダを半角4文字にしておけば，1ロングワードの比較ですむので，ずいぶんと検索が楽になります。

これをプログラムにするにはメモリ管理ポインタの知識が必要不可欠ですから，簡単に触れておきましょう。

自分自身のメモリ管理ポインタは起動直後のa0レジスタによって示されています。ひとつ前のメモリ管理ポインタを知るにはa0レジスタを使って，

```
movea.l  (a0),a0
```

とします。a0レジスタが0ならこれより前にメモリ管理ポインタはありません。プログラムの先頭はメモリ管理ポインタ＋100Hです。つまり，

```
lea.l  $100(a0),a1
```

とすれば，a1レジスタにプログラムの先頭アドレスが入ります。

あとはa1レジスタとヘッダを比較して一致すればすでに常駐しているのですから，メッセージを出力して終了させればいいでしょう。このヘッダチェック部分をプログラムにすると次のようになります。

```
header:dc.b   'X68K'
*これより非常駐部
check: movea.l (a0), a0
tst.l     (a0)
beq       常駐していない
lea.l     $100(a0),a1
move.l    (a1),d1
cmp.l     header,d1
beq       すでに常駐している
bra       check
```

2行目はa0レジスタが0かどうかチェックしているのですが，実際問題として自分と同じプログラムが10000Hより若い番地に常駐しているとは考えられません。ですから，

```
cmpa.l  #$10000,a1
bcs     常駐していない
```

としても問題はないと思います。

紹介したサンプルにはヘッダもないし，ファイル名の検索もしていません。サンプルでは垂直同期割り込みが使われていれば常駐しないようになっています。自分自身が垂直同期割り込みを使っているのでこれでいいのです（手抜きかな）。

常駐プログラムには割り込み処理やベクタの変更がついてまわりますから，これらについても話さないわけにはいきません。割り込みを使うにはIOCSコールを利用します。注意してほしいのは割り込み処理の前後でレジスタの値が変わらないようにすることです。サンプルでもレジスタの保存をしているのがわかると思います。

割り込みを利用してプログラムを呼び出したときは，rtsではなくrteを使います。割り込みは何種類もあるのでひとつずつ説明していくスペースはありません。ぜひプログラマーズマニュアルなどの参考資料で調べてみてください。

次に，ベクタを変更する場合です。ベクタを書き換えるといろいろと楽しいことができます。昔，電脳倶楽部にあったゲボボディスクを覚えている人も多いでしょう。あれはディスクのイジェクト関係の割り込みベクタを変更したものです。ベクタの変更はDOSコール$FF25(_INTVCS)を使って簡単にできますが，不必要にベクタを書き換えることはシステムの破壊にもつながるので，自信のない人にはあまりすすめられるものではありません。

さて，ここまでプログラムを常駐させることに夢中になっていました。常駐できるようになったら解除もできるようにしなくては片手落ちです。これにはDOSコール$FF49(_MFREE)を使います。

```
move.l memptr,-(sp)
dc.w   $ff49
addq.l #4, sp
```

上のように使います。memptrには開放したいプログラムのプロセス管理ポインタを指定します。普通は常駐プログラムを解除するときはスイッチを指定して行うようになっているものが多いのですが，ここでは簡略化して説明しましょう。

先に紹介したヘッダチェックプログラムを使ったとします。すでに同じ常駐プログラムが存在している場合はラベル「すでに常駐している」に分岐します。そのときのa1レジスタはすでに常駐しているプログラムの先頭アドレスを示しています。これよりプロセス管理ポインタはF0H前にありますから，計算によって簡単に求めることができます。割り込みを使っている常駐プログラムの場合は，割り込みを禁止することを忘れてはいけません。「おかしな命令を実行しました」なんてメッセージが出てしまいますよ。

これでプログラムの常駐と解除の方法についてわかってもらえたと思います。X1で同じようなことをするなら，LIMITでマシン語領域を確保して，そこに常駐プログラムを置いておくだけです。割り込みの設定はZ80でもけっこう大変なものがありますが，X68000ではプロセス管理やメモリ管理の話が絡んでくるので相当難しく思えるかもしれません。世にはいろいろと常駐ソフトが出回っているので，デバッガで他人のプログラムを解析してみるのもいい勉強（悪い勉強になる可能性もある）になるかもしれません。Oh!XにもOPMDなどの常駐プログラムがソースリストで公開されているので，それらを参考にするのもいいでしょう。（影山裕昭）

質問にお答えします

日ごろ疑問に思っていること，どんなことでも結構です。どんどんお便りください。難問，奇問，編集室が総力を上げてお答えいたします。ただし，お寄せいただいているものの中には，マニュアルを読めばすぐに回答が得られるようなものも多々あります。最低限，マニュアルは熟読しておきましょう。質問はなるべく具体的に機種名，システム構成，必要なら図も入れてこと細かに書いてください。また，返信用切手同封の質問をよく受けますが，原則として，質問には本誌上でお答えすることになっていますのでご了承ください。なお，質問の内容について，直接問い合わせることもありますので，電話番号も明記してくださいね。

宛先：〒108 東京都港区高輪2-19-13 NS高輪ビル ソフトバンク株式会社出版部「Oh! X質問箱」係

FROM READERS TO THE EDITOR

いまはすっかり夏休み。外では暇そうな子供たちが走り回り，どこへ行っても人ばかり。こっちは仕事だというのに……(単なるひがみ)。まあいいか，会社にいれば涼しいし。しかし，それではあまりにも悲しいのでどっか行こうかな。

◆徹夜明けにYETを3回やって学校へ行く。JRの中で上から3色のブロックが……。"ちゃららん♪"，はっっ。目が覚めると私は終点門司駅。その日私は1限目に間に合わず，学科の全28名に大笑いされ……。また今日も徹夜明けにYETを3回やって。レポートの山に泣いてんだよぉぉッ！　岩瀬　貴代美(18)福岡県

よかったじゃないですか。終点が一応福岡県内の門司駅だったんだから。もしそうでなくて，そのまま関門トンネルを通って山口県の下関まで行ってしまったりしたら，もう大変（そんな電車はないのかもしれない）。

◆17歳のときに兄が買ったファミコンがゲームとの出会い。ゲームの世界にのめり込んで，はや7年。24にもなってこんなことをしていていいんだろうかと思う今日この頃。すべてのゲームが面白いと感じたあの頃に戻りたいと思う私です。　大吉　千恵子(23)茨城県

あの頃に戻りたいというのは，ひょっとして初心に戻りたいという理由ではなく，若返りたいということなのだろうか……。ところで，どっちの年齢が本当なんですか。

◆初めてOh!Xを読んだとき，はっきりいってまるでわからなかった。2冊目……，やっぱりわからなかった。3冊……，4冊……，ずっとわからないまま？　……グスン。と思いきやこの頃少〜しわかるようになってきたみたい。30のねーちゃん（未婚ですのでおばさんと呼ばないように！）になってからでも，お勉強ってできるんですね。がんば！　Oh!Xさん，これからもどうぞよろしく

竹内　夕起(30)静岡県

いえいえ，こちらこそよろしくお願いします。ほかにも内容がわからない人もいるかと思いますが，いきなり全部を理解しようとせずこの方のようにのんびりとわかるところを増やしていけばいいじゃないですか。

◆主人のコンピュータでゲームをするだけなので，7月号の特集さっぱりわかりませんでした。ごめんなさい。　岩崎　利香(20)大阪府

いえいえ，読んでいただくだけで大変光栄です。

◆このハガキを7月1日以降に出すとソフトバンクには届かないのだろうか。

山崎　幹生(16)新潟県

いえいえ，ちゃんと届きましたよ。郵便屋さんが転送してくれますし。転送といってもスタートレックみたいなやつじゃないよ（そんなことは皆さん重々ご承知)。

◆私は某予備校の効きすぎるクーラーのせいか最近体の調子がよくありません。編集室は適温ですか？　田舎のPureな空気とうちわで健康を維持しタメになる記事を書き続けてください。　上原　拓哉(18)滋賀県

僕の出身地は大阪だから，空気はそんなにきれいじゃありません……。でも，田舎っていいですよね。

◆私の部屋のディスプレイCZ-600Dには，RF端子でファミコンが，ビデオ入力端子にはメガドライブが，デジタルRGBにはX1turboZが，そしてアナログRGBでX68000EXPERTⅡがつながっている。いやあ，CZ-600Dって本当にすごいですねえ。デザインもいいし。

梅津　信幸(18)京都府

いいなあ，部屋にものがいっぱいあって。

◆ある日，学校の図書館で僕が「なんでここにはOh!Xがないんだ」なんてひとり言をいうと横にいた友人Kが「あっ。ということはおまえ持ってるのか」と聞いてきた。もちろんX68000のことだと思って「持ってるぞ」「おまえもか！　やっとユーザーが見つかった」と，しばらくの間，2人はとても喜んだ。2人とも初めて同じユーザーが見つかったのだから。しかし，「おれドラスピ持ってるから何かと交換しよう」と言ったら，「へ？　それX68000だろ。なんのことだ。おれX1turboだぞ」。その瞬間2人は白髪と化した……。というわけでX68000はユーザーが少ない。カナシイ……。

福井　達也(16)滋賀県

そんなに悲観するほどX68000のユーザーが少ないとは思わないんですけど。出荷台数も10万台を突破したことだし。まあ，これからどんどん出会うようになるでしょう。

◆天王寺公園がきれいになっていました。編集室には関西出身の方が大勢いると聞きました。大阪出身の方でお盆休みなどで帰省される方は1度行ってみると涼がとれてよいと思いますよ。入園料は150円です。

鯛　富之(28)大阪府

別に関西出身は大勢いないんですけど……。まあ，僕は大阪出身なので暇があれば行ってみます。

◆この間，あの「X68000ステッカー」を買ってきた。何に貼るのかってーと，学校のゲタバコのくつを入れるロッカーに貼るのである。うぉー，目立つ目立つ（カッコいい！)。

清水　了(16)大阪府

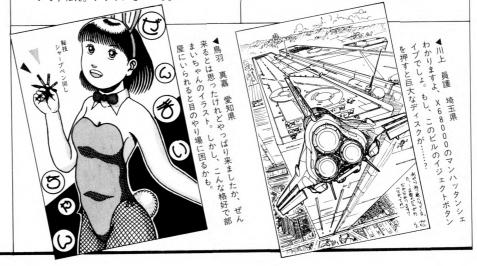
そんなとこに貼らずにおでこに貼るとかしたらもっと目立つかもしれない。

◆うちの卓球部では国体予選で落ちたのでみんなスポーツ刈りにすることになった。夏は坊主頭で過ごすよ。　高橋　政考(16)宮城県

いいなあ，涼しそうで。でも，坊主頭にする気は全然ないけど。

◆今春，うまいことにある外資系の会社の研究室に潜り込んだのだが，毎日英文のレポート

◀鳥羽　真嘉　愛知県
来るとは思ったけれどやっぱり来ましたが，ぜんまいちゃんのイラスト。しかし，こんな格好で部屋にいられると目のやり場に困るかも。

◀川上　員護　埼玉県
わかりますよ，X68000のマンハッタンシェイプでしょ。もし，このビルのイジェクトボタンを押すと巨大なディスクが……？

を読まなきゃならんことになってしまった。強制的に英会話のカリキュラムを受けさせられるわで……。英語なんてどわぃキライだあ～。うっうっ家に帰ればX68000のコマンドは全部英文だしぃ～。ん～ん！
原田　真志(19)静岡県

外資系というとマクドナルドかな？　それとも，ケンタッキーフライドチキン？（決してバカにしているわけではありません）

◆労災で骨折して，はや3カ月。会社にも行けない，友人にも会えない。ましてや，彼女もいない寂しさから，ついX68000を買ってしもうた……。Hソフトばっかやってる毎日。このままではいかーん！比企野　一雄(25)静岡県

本当にそのままではいかーん。

◆X68000SUPER-HDの最大の目玉は丹氏が書いているように「SCSI」の標準装備であろう。しかし，問題がひとつ潜んでいる。SCSIは世界共通規格であるので他社製の周辺機器を使えるようになるのだが，チタンカラーに合わせた，さらにマンハッタンシェイプに違和感のないデザインの機器がシャープ以外から出るのだろうか？　外付けのハードディスクのときもこだわりのXユーザーとして抵抗があったように思うのだが。　山本　雅昭(34)神奈川県

うーん，自分で色を塗るしかないかもしれない。

◆編集部の皆さん，こんにちは。僕はいま目が回るような感動に震えています。Oh!Xのおまけディスクに入っていた「OPMD」を使ったからなのです。実は，以前に打ち込んだOh!X Live in '90のプログラムがサンプリングと同期されてバカスコとスピーカーから流れてきたときには涙がこぼれそうでした。誰しもが，「なんだいまさら。こりゃ！」と思われるでしょうが，本人はそう思われても平気なくらいにはしゃいでます。ううう……。Oh!Xさん，ありがとう！　森田　宣幸(19)宮城県

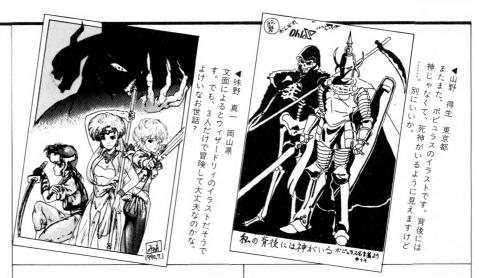
なんだいまさら。こりゃ！（とかいったりして）

◆いまさらながら，X68000の冷却ファンはうるさい。せめて30分動いたら5分回るというようにならないものか。昔，PC-8801mkⅡモデル30のときはディスプレイ付きで4万円だったのをいいことに，冷却ファンの配線を外して使っていたものだ。通気孔にガムテープ貼ったりもした。でも，ぜ～んぜん壊れなかった。まあ，X68000はプラスチック製でいかにも壊れそうだし，値段も張るので無茶なことはやめているが。　三宅　宏典(17)岡山県

たぶん，やめたほうがいいと思います。

◆X140という雑誌を見つけた。だが，実はOh!Xが逆さまになっているだけの話だった。
西本　圭輔(15)東京都

しかし，誰が逆さまにしたんだろう。

◆とうとう親に無断でX68000PROを買ってしまった。早くC言語を覚えてナイトストライカーやメタルサイトを超える3Dシューティングを制作したい。しかし，現実はゲームセンターとなりつつある。はたして，元を取り戻すにはソフトを作って売るのと，友人にゲームのプレイ料を請求するのとではどっちが早いだろうか？　沢井　博(20)宮城県

はたらけー。

◆どうかお許しください。私はあやまってOh!X 7月号をある雨の日に地面に落としてしまいました……。おお！　せっかくの純白が汚れてしまった。　森　義徳(24)大阪府

まさか，わざとじゃあないでしょうね。

◆シミュレーションファンとしては戦争もの以外にも何か出てほしいですね～。X68000用でもいいからスポーツ関係で何かないですかね～。アメフトとかラグビー，ラクロス，オーストラリアンフットボールなどなど。しかし，ルールを知らない人が多いから売りにくいかも。ゴルフゲームをこれ以上リアルにするには経験を積まなければ場所による風の違いがわからなくするとか，自分の打つ前にライバルがバーディを決めてプレッシャーをかけてくれるとかいうのもいいんじゃないでしょうか。　山田　和志(21)奈良県

スポーツなら，やっぱりインディアカとかを出すといいんじゃないでしょうか。

◆透視力の話が出ましたが，たとえ透視力が使えるようになったとしてもピント合わせが難しいのではないでしょうか？　だいたい服なんて1cmもないんですから透視しすぎて2cmぐらい奥を見ようもんなら，体内の内臓とかが見えてしまうと思うんですけど……。ま，医者にとってはカラー版のX線写真として使えて便利なんでしょうけどね……。
佐々木　美武(17)富山県

あー，想像するだけでも気持ち悪い。

◆ひとり暮らしをするようになって明け方までパソコンをやる日々が続いている。普通は体を心配するが，僕は電気代のほうが心配である。パソコンって電気食うのかなあ。
河野　英士(19)愛媛県

そんなに電気は消費しないと思いますが。実は夜食代とかのほうが多かったりして。

◆「ソフトバンク（株）」へ社名変更並びに新社屋完成おめでとうございます。しかも，まったくもっておめでたいことに私は第1種情報処理試験に合格しました。月刊情報処理も買ってただけで置きっ放しだったのだが……。午前問題で出たRISCとかTSSとかはOh!Xで仕入れた知識でいけましたが，問題は関連知識の選択！　数学と英語から4問取ってあと1問，工業から取るつもりが答えがわからなくてその場で金融に変えるという……。よかった，共通1次の倫理政経を3回も受けておいて（笑）。さあて，次はオンラインだ！
広瀬　拓(21)京都府

僕も共通1次は3回受けたけど，あんまり役には立っていないようだ。

◆大学で中国語が第2外国語の中に入っているだろうが，「中国語は簡単だ」なんて思っているあなた！　それはとんでもない大間違いであーる。中国語はとーっても面倒臭ーくて，いやになる言語である。夏休みに中国語の大変な課題が出た！　うわー，泣けてくる。トホホ。　小野寺　光(20)宮城県

ドイツ語もフランス語も面倒臭いよーん。

◆暑中お見舞い申し上げます。東京は水不足で大変だそうですね。水がなくてOh!Xが発売できなくなってもいいですから，無理しないでください。　谷野　育規(18)和歌山県

はい。

◆そーか，そーだったのか。いくら送っても採用されないと思っていたら，僕が送ったディスクはフォーマットされていたのか……。もうこうなったら仕返しだ。もしプレゼントのディスクが届いたらフォーマットして使ってやる！　曽我部　恭昌(15)富山県

そんなこと言うんだったら，プレゼント当ててやんないもーん（子供のケンカか）。

◆編集室へ届いたフロッピーを1枚につき，ノーブランド50円，ブランド品100円で買います。10枚セットならZ80Aに4.7kΩの抵抗3本を付けて交換します。いや，たまたま机の上に転がっていたもので……。ウソだよー！
白井　亮(17)神奈川県

今月号から読んだ人はこの2通のハガキがなんのことだか全然わからないんだろうな。

◆今度ひとり暮らしを始めようと思うのですが，

◀味野　真一　岡山県
文面によるとウィザードリィのイラストだそうです。でも，3人だけで冒険して大丈夫なのかな。よけいなお世話？

◀山野　得生　東京都
またまた，ポピュラスのイラストです。神じゃなくて，死神がいるように見えますけど……。別にいいか。

現在ひとり暮らしをしている人に聞きます。だいたい1カ月いくらぐらいで生活できるのでしょうか。　　　　井田　奨一(22)東京都
それはその人の育ってきた生活レベルによってかなり異なります。貧乏な暮らしをしてきた人は安くて済みますが，豪勢な暮らしをしてきた人は何百万もかかるかもしれない。

◆大学の研究室で使っているPC-9801VXのハードディスクが死んでしまった……。壊れたという話はよく聞いていましたが自分のところへ番が回ってくるとは。あー，自分のX68000のでなくてよかった（といいつつも青くなってバックアップを取った私でした)。
広瀬　真二(24)東京都
番が回ってきませんように……。

◆Oh!Xは若い読者ばかりでなく，高年齢者も読んでいることを知ってください。
合原　宏(61)千葉県
いえいえ，ハガキを読んでいると高年齢者も結構いることに気づきますよ。

◆4月に大学生になって6月よりバイト代をすべて預金しております。計算すると1年後にX68000 SUPER-HDが買える予定。あくまでも計算上であるところが悲しい。
小川　佳昭(19)東京都
計算すると200万年後ぐらいに都内に家が買える予定。

◆今日（7/8)，昼前にNHKを何気なく見ていたら，いきなりソーサリアンのエンディングが流れてきた(たぶんそうだと思う)。アザラシが泳いでいるシーンだった。NHKでもゲームミュージックを使うのかと驚いてしまった。
宮岡　三幸(23)千葉県
ふーん。

◆現在私のX68000はイカれています。電源を入れるとOSとかゲームは走るんだけれど，画面がウルトラQしてしまいます。PC-8801をつないでみたら異常ないのでイカれているのはX68000本体のほうだと思う。そんなわけで私は毎日祈るようにしてスイッチをオンにしています（早く修理に出せばいいのにね)。
新帯　達哉(19)愛知県
いいなあ，ウルトラQがいつでも見られて（そういう問題じゃなーい)。

◆美しいものはやっぱり自然に生きる生物だと思う。表紙にはアッと誰もが思うような生物のグラフィックを載せてもらいたい。いままでの表紙は世の中や社会に掃いて捨てるほどあるようなパターンばっかり。うんざり。うんざり。もーいや。やめて。心が乾く。潤いを！　　　　新畑　貴史(19)広島県
そうですよね。オコゼとか，チョウチンアンコウとか，ナマコとか，サナダムシとか，自然には美しいものがいっぱい（僕ってやなやつでしょ)。

◆X68000のウイルスで大騒ぎしてましたけど，Macintoshでは日常茶飯事のようです。ゼミの予習で使っていたら画面に死神が出てきました。ゼミは明日だというのに。誰か私の予習ディスクを元に戻してください。
加藤　弓弦(22)大阪府
じゃあ，あなたのディスクを元に，つまり生ディスクに戻してあげましょう。

◆マシン語，CG，ハードウェア工作。やりたいことはたくさんあるのになかなか手が出ない。いままでヒマがないからできないと思っていたが，もしかしたら，そうではないのではないか。足りないもの，それは「締め切り」?
村尾　遊(26)岡山県
「締め切り」があるからといっても，なかなかできないことはよくあります（経験者は語る)。

◆ついにX68000を買った。感動！　ちなみにこいつとMZ-2500が6畳一間の部屋に並んでいるさまはなかなか壮観です。なんと情報処理の1種に受かってしまった。感激！　と幸せいっぱいの日々を送っている最近の私ですが，さすがにX68000のン十万の借金はつらい……。　　　　中村　祐一(20)東京都
楽あれば苦あり。起きて半畳，寝て1畳。

◆MIDIボード買ったけど，音源まで金が回らないという間抜けなあなた（んな人はいないって?)。ローランドの「DR-550 (32,000円)」を買いなさい。これはドラムマシン（いわゆるドンカマってやつ）だから，当然ドラムの音しか出ないけど，これでスーパーハングオンやった日にゃあ，も一最高ですって！　But モトスだと間抜けだけれど……。
野田　敏之(19)神奈川県
だそーです。

◆この間読んだ新聞によると，2030年頃には平均気温が温室効果で2～3度上がるそうだ。そして，夏はいまよりも猛暑で天候不順になるという，となるとパソコンには辛いことになるのは間違いないわけで，熱と雷に対する方法を何か考えねば！（全国の皆さんはどうしているのだろう?)　　岡部　誠(25)福井県
それよりもクーラーを持っていない人（つまり，僕のこと）に辛いことは間違いないわけで，それに対していい方法を考えてもらわねば！　えっ，クーラーを買えって？　そりゃ，もっとも。

◆この夏，ロシア語聴講者と講師でソビエトに旅行に行くはずだった。でも，貯金が足りなかったのであきらめた。いまにして思えばX68000を担保にしてでも行ってみたかった。トホホ……。　　　　神生　直敏(20)栃木県
うーん，ソビエトに行くと涼しいかもしれないなあ。

◆大学生って忙しいものだったんですね。クラブ活動，コンパ，ボウリング，ハイキングにキャンプ，自動車学校にアルバイト，これにパソコンが加わるとまさに体が2ついるなあ。なに？　勉強？　テスト直前まで忘れていなさい，そんなもんは。越智　一秀(19)広島県
はい，私もテスト直前まで忘れているほうでした。別にそんなに忙しいこともなかったんですが。

◆6月9日(土)，ビジネスショウで見たPC-286 Bookのコマーシャルがテレビで。「あれっ，もう売り出していたのか」。6月11日（月)，会社が暇なのをいいことに抜け出して秋葉原へ。ああ，ついにPC-9801互換機を買ってしまった。ゲームをやりたいばっかりに！　遥かなるオーガスタをやる。うん，速い。ポピュラスをやる。反応がよい。ソーサリアンをやる。音がさみしい……。しかし，4kgを超えるとやはり重い。ACアダプタやフロッピー，マウスなんかも持っていかなくてはならんとなるとつらい。で，今週は会社に置きっ放しなのだった。ゲームをするために買ったのに，実は仕事に使っている。X68000のときとは逆だ。
森　啓泰(23)東京都
今週だけでなく，ずっと会社に置きっ放しのような気もする。

◆X6800の100万台突破のためにも，これからもガンバッテください。X6800が国民機になる日も近い！　この前，伝言板にX6800のパンフレットを貼って「これ買わなきゃ損するよ」と書いておいた。　　木下　剛(16)神奈川県
あのー，それはいいんですけど。X6800じゃなくてX68000なんですが。まあ，よくある間違いってやつですね。

◆7月号の特集の中のマルチタスクへの挑戦で

◀尾澤　宏　兵庫県　全体の雰囲気はとてもいいんですけど，よく見ると剣士の手がなんか変。何かを後ろに隠しているように見える。なんだろう，気になるなあ。

◀江副　滋　東京都　本当に久々に登場の江副さんのイラストです。2年振りくらいですか。その間，インドで修業でもしていたんでしょうか。

は昔のことを思い出してしまいました。7,8年前，月刊マイコンにMZ-80シリーズ用のMONIOSとかいうのの記事があって，8253を使ったマルチタスクのお話で，当時MZ-80K2Eを持っていた私はアセンブラでマルチタスクをして遊んだことがありました。

野村　正文(21)茨城県

マルチタスクというと聖徳太子(?)。

◆ナポレオンが日本で人気があるように，東郷平八郎はフィンランドでは人気があるのです。理由は東郷元帥がいつも自分たちフィンランドをしいたげてきた帝政ロシアのバルチック艦隊を日本海海戦で破ったからです。

新熊　昴(34)大阪府

というわけで，7月号の山崎さんの問いかけに対する答えをいちばん早く書いてきてくれたのがこの方です。ああ，ありがたやありがたやと思っていたら……。

◆たしか日露戦争で当時世界最強と謳われていたロシアのバルチック艦隊に弱小日本の帝国艦隊が勝ってしまったということで，その頃ロシアに支配されていたフィンランドの人々が大いに勇気づけられた。ということなのだそうです。だから，向こうでは「東郷のヘーチャンは英雄なんだ！」って高校の時の英語の先生(6?歳)が言ってました。

伊藤　義博(19)東京都

◆たとえ，いまソ連が変わってきたといっても過去に数え切れない死傷者と政治的圧迫で大ダメージを受けたトルコ，フィンランド両国は歴史上でただ1国ロシアを屈服させた日本の東郷平八郎を称えて，それぞれ彼の像，将軍ビールシリーズというものを作って心中うっぷんをはらしているのです。

上田　秀晃(19)福島県

◆いまを遡ること85年，フィンランドは帝政ロシアの支配下にありました。しかし，日本海海戦でバルチック艦隊を東郷平八郎率いる日本軍が破り，ロシアは戦争に負け，ロシア革命が起こり，フィンランドは独立を勝ち取ることができたのです。フィンランド人というのはなんと義理堅い人たちではありませんか。

越智　亮(17)島根県

◆書いてきた人も多いかな？　フィンランドの缶ビールはアドミラル（将軍）ビールといって，東郷平八郎がロシアの艦隊を破ったとき，当時ロシアに苦しめられていたフィンランドがその勝利を称えて作ったものらしいですよ（山崎さんへの回答をする山崎）。

山崎　勇(28)新潟県

◆週刊朝日百科「日本の歴史」（朝日新聞社）によると世界海戦史上で「The Battle of Tsushima」の名で呼ばれる日本海海戦の指揮官として大胆なる敵前大回頭をとった勝利の見事さによって，東郷平八郎は「東洋のネルソン」として喧伝された。そのため，バイキングの故地ともいうべきフィンランドでネルソンとともに著名な提督の名をつけたビールの商品名になり，現在も売られているそうです。

青山　和広(18)愛知県

と，このように山のような東郷平八郎のハガキが送られてきて（ここに載っているのはそのうちのごく一部），最近までその日に来たハガキを読んでいくと必ず何通かは東郷平八郎に関するハガキを目にするという日々が続きました（東郷平八郎の5文字が頭にこびりついて離れないよー）。どうもたくさんのハガキありがとうございました。これで山崎さんもぐっすりと眠ることができるでしょう（いままでもちゃんと眠ってた？）。

▲住友　智代　愛媛県
再び登場の住友智代さんです。女性の投稿は少ないので，がんばってください。しかし，ダンジョンマスターのイラストとは珍しい。

ぼくらの掲示板

●掲載ご希望の方は，官製ハガキに項目(売る・買う・氏名・年齢・連絡方法……)を明記してお申し込みください。
●ソフトの売買，交換については，いっさい掲載できません。
●取り引きについては当編集部では責任を負いかねます。
●応募者多数の場合，掲載できない場合もあります。

仲　間

★X68000ユーザーズクラブ「WATER」では第2次会員を募集しています。今回より「HYPER-SHAKE」というディスクマガジンを発行しております。X68000を持っているあなた！　1度連絡ください。損はさせません。入会資料のみ希望の方は62円切手を貼った封筒を，ディスクマガジンを見てみたい方は400円分の為替を同封のうえ，下記までご連絡を。〒975　福島県原町市小川町109-2　山崎潤一(21)

★X68000ユーザーのための，X68000ユーザーによるサークル「兎団」ではただいま会員を大募集しています。活動は主にプログラムや情報などの交換をしたり，シェアウェアソフトの発表をしたりします。合言葉は「打倒電脳倶楽部」です。やってみて損はありません。詳しくは120円切手3枚同封のうえ，下記までご連絡を。〒503-21　岐阜県不破郡垂井町宮代2840-1　田川和義(16)

売ります

★カワイKIII（箱以外付属品はすべてあり，新同品)をRAMカード，ROMカードを各1枚付けて5万5千円で。希望によりソフトケース（3千円増)，スタンド（5千円増）も付けます。連絡は往復ハガキで。〒243　神奈川県厚木市妻田1,158そりだハイツ1,423　山野和也(19)

買います

★MIDI音源ローランド「CM-64」を6万〜7万円，シャープ「CZ-6BM1」か「SX-68M」を1万円以内で。いずれも完動，箱，マニュアル，付属品付き。連絡はハガキで。〒164　東京都中野区東中野2-3-8林方　安嶋栄祐(19)

★CZ-6VT1を3万円で。多少汚れていたり傷があっても結構です。連絡は往復ハガキで。〒186　東京都小平市上水南町4-9-8　山野得生(18)

★カラーイメージボード「CZ-8BV2」または「CZ-8BV1」を送料込み1万円で。連絡は往復ハガキで。〒652　兵庫県神戸市兵庫区塚本通3-1-4　岡崎礼子(25)

★X68000の増設RAMボード「CZ-6BE1」を1万5千円前後で（マニュアル付き）。連絡はハガキで。価格優先で1人だけに2週間以内に連絡します。品物が届いてから代金を送るつもりですので，その条件では都合の悪い方はご遠慮ください。〒399-02　長野県諏訪郡富士見町富士見4654　宮下朗(18)

★X1用FM音源ボード「CZ-8BS1」（付属ソフト，マニュアル付き）を8千円で。完動品，外観不問。1987年10,12月号，1988年12月号および1989年1,2月号を付けてくれる方には1万1千円で。連絡は往復ハガキで。〒839-11　福岡県久留米市善導寺町与田37-14　鹿田剛(18)

バックナンバー

★Oh!X1989年1〜8月号を送料込み6千円で買います。切り抜き不可。連絡はハガキで。〒280　千葉県千葉市星久喜町1173　佐久間優(18)

編集室から from E・D・I・T・O・R

DRIVE ON

このコーナーでは，本誌年間モニタの方々のご意見を紹介しています。今回から新しいモニタの方々にご意見をいただくということで，記事に関することだけではなく，コンピュータや本誌に関していろいろなことを質問させていただきました。これからもいろんなことを聞かせてもらいたいと思います。1年間どうぞよろしくお願いします。

パソコンについての考え

●パーソナルコンピューティングとは本来「それ自身」を楽しむもの。つまりコンピュータに触りイマジネーションを高めていくところに楽しさがあると思う。また，僕がコンピュータを知った頃というのはそのような考え方が当然の時代であった。BASICにしろ，マシン語にしろ，数々の言語にしろ，それを使って何かものを作る，ということは二の次であるような楽しみ方をしている。考えて，ぶつかって，反応が返ってくるのが面白いとでもいうのであろうか。また，そのような楽しみ方があるからこそ，MZの古いテープベースの言語なども出回ったのであったと思う。

長谷川敦士(17)　MZ-2521，MSX2　山形県

●私はハード屋さんですので，「人間の生み出した技術は人間の生命維持に役立たなくてはいけない」という基本理念を持っています。私が自ら作り出すものにおいても，作る人間，使う人間双方にその有効性を享受しなければいけないと思っています。私の周りの機械たちは人間の害になるものについてはどんなことがあっても排除されるというたいへん厳しい環境に置かれて苦労しているようです（でも，動かなくなったら心を込めて修理します)。しかし，これを少し飛躍した解釈で臨むなら，「人間が生み出した技術は人間が便利になるならどう用いても構わない」ということになると思うのです。言い換えれば，「骨までしゃぶる」ということになります。コンピュータ，とりわけパソコンは信じられないような進化を遂げ，より高い性能を，より使いやすい環境で我々に提供しています。ユーザーが「あぁ，便利だなぁ」と思えばそれはパソコンがその能力を発揮していると考えるべきで，どうして便利と思われたのかは特に問われるものではないと思うのです。つまり，せっかく個人で所有できるのだから，使いたいように使って便利に，そして幸せになればそれでいい，これが私のパーソナルコンピューティングに対する考えです。そもそも私は，コンピュータを個人で所有することなど考えてはいけなかった時代の人々が現実に個人所有できる“コンピュータ”を目の当たりにしたとき，そのギャップの大きさゆえに「パーソナル」と誇張するに至ったのだと考えていまして，あまり個人用ということに対してはこだわりたくないというのが本音だったりするのです。

浅野憲(19)　X68000PRO，X1turboIII，X1F model20，MZ-80C，PC-6001，M5jr，PC-1245　大阪府

最近の本誌の傾向について

●最近特に気づくことであるが，内容がやさしすぎるものがあると思う。記事ではなく初心者に対してこのことをいいたいのである。以下のことを例として挙げるが，これらのことは絶対に人に聞くよりも自分でマニュアルなどで調べて理解しないと，将来本当に理解できないだろうと思う。アフターケアのconfig.sysの書き換え，メモリの件などである。この2つは(ほかにももっとありますが)システムを理解するうえで逃げられないものであるだろうし，人に聞く前に自分で工夫してみてもいいんじゃないかと思う。やはり，年1回か2年に1回くらい，永久保存版でこれ以上やさしくできないというくらいの特集を組むしかありませんね。……でも，言わせてもらっちゃうとマニュアル読んでできることは説明しなくていいと思うし，初心者の甘えはよくないと思う。

中川比呂志(19)　X68000，X1Cs　東京都

特集「マシン語への第一歩」について

●いやいや，私にはむずかしすぎて。でも，「ぜんまいちゃん」は面白かったです。私は「ぜんまいちゃん」の過去を知りませんが。こういった取っ付きやすい記事は大歓迎です。6月号の付録ディスクのPASCALコンパイラで1から100まで順に足すというなんとも単純でアホらしい（しかし，昔はソロバンでよくやったものである。それに，私はそれくらいのプログラムしか組めないのである）プログラムを組んでコンパイルしたのですが，AS.Xがなくて（リンカはHLKを持っている）ソースファイルしかできなかったんですね（P.132の「Q」は私ではありませんが，私もそう思ったもののひとりです)。そこでなんだか興味がわいて，できたソースファイルをEDに読み込んだんです。そこで，「あー」。10行にも満たなかった稚拙なプログラムがなんとも長くわからないものに（当たり前か）なっていたんですね。そこでひと言，「マシン語とはなんと奥深いものなのだろう！」ということで，いつか68000のマシン語を勉強してみたいですね。

安井百合江(16)　X68000PRO　愛知県

●7月号の特集でよかった点は，マシン語がまったくわからない人でも読めるということ。吉田幸一氏の「ぜんまいちゃん」はマシンとマシン語の関係を理解しやすく，しかも面白く説明している。そして，毛内俊行氏の記事ではもっと具体的にアセンブラの使い方を示している。この中で目新しいのは例題があること。たったこれだけのことで，知識としてだけでなくビジュアルに捉えることができる。私もこの例題にはお世話になりました。しかし，そのあとはMC68000の話になってしまって，基礎知識がまったくない私にはさっぱりわからなかった。なお，最後に山田純二氏がしっかりとX1，MZユーザーをフォローしているのを見て，これがOh!Xという雑誌の本来の姿だと思った。

畑剛志(18)　X1turbo model10/ZII，JR-100，MSX　北海道

ごめんなさいのコーナー

8月号　表紙ぎゃらりぃ

1989年7月号の写真が8月号のものになっていました。お詫びいたします。

8月号　XROT0.X

大きなオブジェクトとリンクしたときにリンカのオーバーフローエラーになることがありますので，XROT0.Sの208行のBSR W_LINEをJSR W_LINEに変更してください。また，XROT0.Sは画面サイズを設定する引数W，Hを128までしか受け付けません。

7月号　ノーマルX1への対応

P.140　リスト2の1040行の「π　1」は正しくは「KMODE　1」です。

バグに関するお問い合わせは
☎03(5488)1311(直通)
月〜金曜日16：00〜18：00

お問い合わせは原則として，本誌のバグ情報のみに限らせていただきます。入力法，操作法などはマニュアルをよくお読みください。

また，よくアドベンチャーゲームの解答を求めるお電話をいただきますが，本誌ではいっさいお答えできません。ご了承ください。

港区高輪産 Oh!X第1号 それがどうしたって?

▼編集室が引っ越し，今月号は新しいビルで初めて作ったOh!Xとなります。一口に引っ越しといっても，Oh!MZ時代からのマシンやソフト，資料などを引きずるOh!Xの荷物は生半可なものではありません。たった4人の部であるにもかかわらず，荷札の数は500枚を超え，数ある編集部のなかでも最高でした。

その新しい建物ですが，1，2階が銀行で，3〜10階がソフトバンク，最上階がレストランというディレクトリです（ちなみにOh!Xは3階にあります)。新築で設備もなかなかなのですが，なにしろ下に銀行があるものでビルの管理が厳しい。24時間得体の知れないスタッフが出入りする編集部にはきついのです。深夜来るスタッフには駅前の公衆電話から「開けてコール」をかけてもらい，1階の通用口まで迎えに行くという情けなさ。なんとかしてもらいたいのですが。おまけにこの辺には食べるところが異常に少ない。上のレストランは高すぎて行けないし……，編集スタッフの食生活はどうなるのでしょう。

というわけで今回は，Oh!MZの創刊第2号を3名の方に差し上げたいと思います。ご希望の方は綴じ込みのアンケートハガキのプレゼントNo.に0と記入してお送りください。

▼さて，今月号の特集Iは，かねてより近々やらねばと考えていた日本語処理に関するものです。Oh!Xが考えるパーソナルマシンはそれ自体が魅力的であると同時に，ユーザーが使う気になれば実務にも耐えうるものでなければなりません。でも，本誌読者の皆さんの大多数は，年賀状や簡単な文書以外に，日本語の文章を自分で考えて作成する機会があまりないのではないでしょうか。ビジネスソフトが少ないという前に，そういうソフトを必要とする状況を現ユーザーのなかからも生み出していかなければと思うのです。できれば，パソコンで日本語を処理することに関する皆さんのご意見をお聞かせください。

▼今月のINTEGRAL XIは編集の都合によりお休みさせていただきます。

投稿応募要領

- 原稿には，住所・氏名・年齢・職業・連絡先電話番号・機種・使用言語・必要な周辺機器・マイコン歴を明記してください。
- プログラムを投稿される方は，詳しい内容の説明，利用法，できればフローチャート，変数表，メモリマップ（マシン語の場合）に，参考文献を明記し，プログラムをセーブしたテープ（ディスケット）を添えてお送りください。また，掲載にあたっては，編集上の都合により加筆修正させていただくことがありますのでご了承ください。
- ハードの製作などを投稿される方は，詳しい内容の説明のほかに回路図，部品表，できれば実体配線図も添えてください。編集室で検討のうえ，製作したハードが必要な場合はご連絡いたします。
- 投稿者のモラルとして，他誌との二重投稿，他機種用プログラムを単に移植したものは固くお断りいたします。

あて先

〒108 東京都港区高輪2-19-13 NS高輪ビル
ソフトバンク出版部
Oh!X「㋢㋮㋴」係

SHIFT・BREAK

▶まぶたを閉じると，グラデーションが浮かび，耳をすませばジェノサイドの音楽が……。最後の夏休みだというのに暇をもてあましている。楽しみにしていた旅行も行けなくなってしまったし。ここは，涼しいマシンルーム，部屋に帰れば30度を振り切りっぱなしの温度計が僕を迎えてくれる。不毛な日々が続いているなあ。がんばろ。 (純)

▶車で編集室にくると，そのままどっかへ行くこともままある。この前はS.K.氏とタイヤを鳴らして深夜の17号を爆走(でも軽なの)。まあS.K.氏との爆走ドライブはいいとしても，編集のA氏との迷走ドライブには困るなー。なぜかA氏が横に座ると必ず迷うのだ。車の中でダジャレ勝負をしているせいではない，と思うけど。多分。 (H.U.)

▶リンドバーグのボーカルの渡瀬マキが，元アイドルだとは知らなかった。一本取られたな。さて，夏休みはなにをしよう？ 友達はみんな田舎に帰るしなあ（下宿にはクーラーがないが，実家にはあるから帰るのだそうだ)。一方，東京に残る組は冬のスキーに備えてバイトするのだそうだ。まったくなんなんだ？ こいつら。 (亀)

▶出張でスペインとポルトガルに行くことになったのだが，出発予定日の3日前になっても飛行機が何時に成田を出るのか知らされていない。それどころか，旅行会社は昼出発と言い，航空会社は夜出発だと言い張る。おまけに，旅行会社から聞いたポルトガルのホテルの住所は実在しない地名だという噂だ。果たして無事たどり着くのだろうか。 (K.M.)

▶全国的に恥をさらすようだが，いま免停中である。この本が出るころには更生していることだろう。常々，あんな取り締まりで違反者が減るはずないと思っていた。しかし今日，その考えを変えた。あの2日間もの講習はイヤガラセとしか思えない。あれをもう一度受けることを考えると，違反する気が失せる。うーん素晴らしい抑止効果。 (A.T.)

▶効きが悪い→たまに効かなくなる→たまには効く→滅多に効かない→効いたら奇跡，と5段活用してインタラプトスイッチが死んだ。デバッグ中にバコバコ押しまくったからなー。追い打ちをかけるように，今度はリターンキーがチャタり出す。期せずしてCTRL＋Mの人となった僕は，右手小指が楽だぜとうそぶくのだった。 (Mu)

▶そんでもって，ハンニバル・レクターである。人間を論理的に面白く書けるトマス・ハリスって凄いと思う。ハヤカワ文庫NVの「レッドドラゴン」と新潮文庫の「羊たちの沈黙」を読んでちょ。論理的な親切とか論理的な愛とか論理的な狂気とか論理的なギャグとか論理的な極悪非道とか。論理的冷酷さと非論理的情動はやはり必要だよね。 (K)

▶ふと思い立って車を買うことを決心した。契約するのに三文判ではカッコ悪いと思って新しいハンコも作った。だがディーラーは僕に駐車場がないという理由でハンコを押させてくれない。せっかくの意気込みを打ち砕かれた。サービスで駐車場くらい探してくれてもいいじゃない。ああ面倒臭いなあ。
(桃伝ターボの3倍モードが快適なKO)

▶電車に乗っていると，突然，横に座っていた外国人が話しかけてきた。なにかと思ったら自分の降りるべき駅が何駅目かと……そこでとりあえず教えてあげると，お礼に(?)名刺をくれた。名前を見てインド人かと思い「インディアン？」と聞いてみたが見事外れてしまった。今度パキスタンに行ったら家に泊めてね。モハメッド＝ユーナスさん。 (A)

▶編集部の引っ越しを機に，杉並区民から横浜市民になる決意をした。が，問題は荷造り。4年間で蓄積されたおおよそ独り暮らしとは思えない物量をまとめるのは不可能に近い。何人か友達を引っ張り込もうかとも思ったが，また不特定多数の人間が出入りしそうなのでやめた（今度は3LDKだし)。ああ，ドラえもんがほしい……。 (と思う怠け者のE.O.)

▶「ASK68Kってなにがまずいんでしたっけ？」「やっぱり単漢字じゃないですか」「読みがなも単漢字登録すればよくなりますよ」「なるほど」「ほかにまずい点って？」「なんでしたっけ？」現在のところメイン辞書740Kバイト，サブ辞書118Kバイト。もう少しなら，なんとかなるな。あとはレーザープリンタもほしいところだが……。 (U)

▶8つの編集部が1つの大きなフロアに並んだ。整然としたオフィスの端に立つと，1カ箇所だけニョキニョキとそびえ立つ棚の上にさらに荷物が積み上げられた火山地帯が視界をさえぎる。そこが秘境と呼ばれるOh!XとOh!FMの編集部だ。しかも夕方になると掃除機を背負ったゴーストバスターズおばさんがやってくる。「ここって迷路ね」だって。 (T)

microOdyssey

我々にとってワープロは商売道具だ。原稿用紙を相手にする代わりにワープロで文字を追う，それが当たり前になって3,4年。すっかりワープロに慣れきってしまった。いまでは原稿用紙にひとマスずつ字を埋めて論理的文章を完成させるということは神業のようにも思える。どうして昔はそんなことができたのだろうか。

ワープロは文書作成効率を劇的に引き上げる。アイデアプロセッサという考え方が出る以前からワープロそのものが同種の機能を果たしていたからだ。

が，弊害もある。それは文字フォントのせいかもしれないし，一度にアクセスできる情報量のせいかもしれない，とにかく画面上ではケアレスミスが多くなる。デバッグ効率同様，一度プリントアウトしたものと画面上のものでは認識のしやすさが違うのだ。結果として，ワープロで書いた文章は小さなミスが発生しやすい。また，安易に切り貼りすることができるためワープロゆえの悪文というのも出てくる。下書きがそのまま清書となりうるからか，つい推敲が甘くなる。

そう，ここで問題なのは「下書き用」のワープロなのだ。ここでいう下書きとは，文書の内容そのものだ。ワープロはよりよい内容を書くことをサポートするツールであってほしい。ことワープロにおいて「清書用」という言葉が悪口以外で使われないのは，下書きの段階を無視して清書を行おうとするものがあまりに多いからだ。下書きができない状態でDTPについて語っても実のあるものにはなるまい。ワープロに関する最初の問題点といえるだろう。

日本語処理の最終的な問題点に入力装置を挙げる人は多い。そして，多くの人が手書き文字認識，音声入力などに期待をかける。

手書き文字認識は一部の電子文具ですでに実用化されている。ひらがなを入力して変換機能を併用するもの，直接漢字を認識するもの。認識精度や速度はいくらでも改善可能だから特に問題にはならない。字画数が多くなると手書きが本当に楽なのかという疑問は残るが，これも技術が解決してくれる可能性がある。まあ，期待してみるのも悪くない。

対して，音声入力だ。入力のための労力はさらに軽減されるだろう。が，音声入力をしている図を想像するとなかなか不気味なのではないか，という意見もある。確かに音声入力ワープロでラブレターを書く男というのはあまり想像したくない。

たとえ，完成したら公になるものだとしても，書きかけの文章を他人に覗かれるのは気持ちのいいものではない。まして，プライベートな文書ならなおさらだ。音声入力が一般的な入力装置として確立される際にこういった問題は解決されるだろうか？

そうこうしているうちに，いっそ「思考を直接入力できたら……」という夢物語に行きつくことになる。頭に浮かんだことが，淀みなくディスプレイに表示される画期的インタフェイス。あらゆる点で理想的だといえよう。夢物語？　手首に電極でも埋め込む気かって？　ところが，それに非常に近いものはすでに存在している。そう，それは「ブラインドタッチ」とも呼ばれている技術だ。(U)

1990年10月号 9月18日(火)発売

特集　電子音楽術入門
ASK68K用辞書ユーティリティ後編
詳報C compiler PRO-68K ver.2.0
全機種共通システム
ライブラリアンWLB
CARD.FNC用ゲーム
ひとり占いTEN

バックナンバー常備店

地域	地区	店名・電話
東京	神保町	三省堂神田本店5F 03(233)3312
	〃	書泉ブックマートBI 03(294)0011
	〃	書泉グランデ5F 03(295)0011
	秋葉原	T-ZONE 7Fブックゾーン 03(257)2660
	八重洲	八重洲ブックセンター3F 03(281)1811
	新宿	紀伊国屋書店本店 03(354)0131
	高田馬場	未来堂書店 03(200)9185
	渋谷	大盛堂書店 03(463)0511
	池袋	リブロ池袋店 03(981)0111
	〃	西武百貨店9F コンピュータ・フォーラム 03(981)0111
神奈川	横浜	有隣堂横浜駅西口店 045(311)6265
	〃	有隣堂ルミネ店 045(453)0811
	藤沢	有隣堂藤沢店 0466(26)1411
神奈川	厚木	有隣堂厚木店 0462(23)4111
	平塚	文教堂四の宮店 0463(54)2880
千葉	柏	新星堂カルチェ5 0471(64)8551
	船橋	リブロ船橋店 0474(25)0111
	〃	芳林堂書店津田沼店 0474(78)3737
	千葉	多田屋千葉セントラルプラザ店 0472(24)1333
埼玉	川越	黒田書店 0492(25)3138
	川口	岩渕書店 0482(52)2190
茨城	水戸	川又書店駅前店 0292(31)0102
大阪	北区	旭屋書店本店 06(313)1191
	都島区	駸々堂京橋店 06(353)2413
京都	中京区	オーム社書店 075(221)0280
愛知	名古屋	三省堂名古屋店 052(562)0077
	〃	パソコンΣ上前津店 052(251)8334
	刈谷	三洋堂書店刈谷店 0566(24)1134
長野	飯田	平安堂飯田店 0265(24)4545
北海道	室蘭	室蘭工業大学生協 0143(44)6060

定期購読のお知らせ

Oh!Xの定期購読をご希望の方は綴じ込みの振替用紙の「申込書」欄にある『新規』『継続』のいずれかに○をつけ，必要事項を明記のうえ，郵便局で購読料をお振り込みください。その際渡される半券は領収書になっていますので，大切に保管してください。なお，すでに定期購読をご利用の方には期限終了の少し前にご通知いたします。継続希望の方は，上記と同じ要領でお申し込みください。

海外送付ご希望の方へ

本誌の海外発送代理店，日本IPS(株)にお申し込みください。なお，購読料金は郵送方法，地域によって異なりますので，下記宛必ずお問い合わせください。

日本IPS株式会社
〒101 東京都千代田区飯田橋3-11-6
☎03(238)0700

Oh!X 9月号
■1990年9月1日発行　定価560円(本体544円)
■発行人　孫　正義
■編集人　橋本五郎
■発売元　ソフトバンク株式会社
■出版事業部　〒108 東京都港区高輪2-19-13　NS高輪ビル
Oh!X編集部 ☎03(5488)1309
出版営業部 ☎03(5488)1360 FAX 03(5488)1364
広告センター ☎03(297)0181
■印　刷　凸版印刷株式会社

ソフトバンクの書籍特約書店

下記の書店の一覧は、ソフトバンク書籍特約店として右にある商品の他、新刊もとりそろえております。ご希望の商品がある場合は、下記のお近くの書店にてお買い求め下さい。

(注)現品が売れて補充中の場合もございますので、ご注意下さい。

SOFT BANK
ソフトバンク出版事業部
〒108 東京都港区高輪2-19-13 ☎03(5488)1360

全国特約書店一覧

地域	書店名	電話番号
〈北海道〉		
札幌市	紀伊國屋書店札幌店	011-231-2131
〃	旭屋書店札幌店	011-241-3007
〃	丸善らがぁーる新札幌DUO店	011-890-2588
〃	冨貴堂札幌パルコ店	011-214-2303
〃	ダイヤ書房本店	011-712-2541
〃	ダイヤ書房西店	011-665-6223
旭川市	旭川冨貴堂	0166-26-3481
〃	ブックス平和マルカツ店	0166-23-6211
苫小牧市	旭屋書店苫小牧店	0144-36-5185
〈東北〉		
青森市	成田本店	0177-23-2431
〃	岡田書店	0177-23-1381
弘前市	紀伊國屋書店弘前店	0172-36-4511
〃	今泉本店	0172-32-2231
〃	メディアイン城東店	0172-27-8118
八戸市	伊吉書院	0178-44-1917
盛岡市	さわや書店	0196-53-4411
〃	第一書店	0196-53-3355
仙台市	金港堂	022-225-6521
〃	金港堂ブックセンター	022-223-0979
〃	アイエ書店駅前店	022-264-0718
〃	丸善仙台支店	022-266-1127
〃	髙山書店	022-263-1511
秋田市	三浦書店	0188-33-8131
山形市	八文字屋	0236-22-2150
福島市	岩瀬書店コルニエツタヤ店	0245-21-2101
〃	博向堂	0245-21-1161
郡山市	東北書店	0249-32-0379
いわき市	ヤマニ書房本店	0246-23-3481
〃	鹿島ブックセンター	0246-28-2222
会津若松市	宝文館	0242-27-5198
原町市	文芸堂	0244-22-1720
〈関東〉		
水戸市	川又書店駅前店	0292-31-0102
〃	ツルヤブックセンター	0292-25-2711
勝田市	武石書店	0292-73-1212
東海村	大野書店	0292-82-2098
鹿島郡	なみき書店	0299-96-1855
土浦市	共栄堂	0298-21-6134
つくば市	丸善筑波大学会館店	0298-51-6000
〃	友朋堂吾妻本店	0298-52-3665
宇都宮市	落合書店オリオン店	0286-34-3777
〃	落合書店東武ブックセンター	0286-34-8271
〃	新星堂宇都宮店	0286-33-2337
小山市	進駸堂駅ビル店	0285-25-1522
前橋市	煥乎堂	0272-23-1211
〃	リブロ前橋店	0272-34-1011
〃	戸田書店前橋店	0272-61-5063
高崎市	学陽書房	0273-23-4055
〃	サカヰ書店	0273-62-1500
〃	新星堂高崎店	0273-27-3961
〃	戸田書店高崎店	0273-63-5110
太田市	ナカムラヤ	0276-22-2001
〈首都圏〉		
浦和市	須原屋本店	048-822-5321
〃	須原屋コルソ店	048-824-5321
大宮市	押田謙文堂	048-641-3141
〃	ブックセンター押田	048-647-3141
〃	三省堂ブックポート	048-646-2600
蕨市	須原屋蕨店	0484-44-1211
川口市	岩渕書店川口店	0482-52-2190
川越市	黒田書店川越店	0492-25-3138
所沢市	芳林堂所沢店	0429-25-5355
〃	いけだ書店所沢店	0429-28-3271
上福岡市	黒田書店上福岡店	0492-66-0120
朝霞市	文教堂朝霞店	0484-76-0107
志木市	新星堂志木店	0484-74-0182
春日部市	文教堂春日部店	048-752-7666
比企郡	錦電サービス	0492-96-2962
千葉市	多田屋セントラルプラザ店	0472-24-1333
〃	キディランド千葉店	0472-25-2011
習志野市	巌翠堂	0474-72-5011
船橋市	ときわ書房本店	0474-24-0750
〃	リブロ船橋店	0474-25-0111
〃	旭屋書店船橋店	0474-24-7331
〃	芳林堂津田沼店	0474-78-3737
〃	第二巌翠堂	0474-65-0926
〃	三省堂書店西船橋店	0474-34-3111
柏市	西口アサノ	0471-44-2111
〃	新星堂柏店	0471-64-8551
松戸市	堀江良文堂	0473-65-5121
〃	辰正堂駅ビル店	0473-64-7997
横浜市	有隣堂トーヨー店	045-311-6265
〃	有隣堂東口ルミネ店	045-453-0811
〃	栄松堂相鉄ジョイナス店	045-321-6831
〃	そごうブックセンター	045-465-2111
〃	丸善ブックメイツポルタ店	045-453-6811
〃	有隣堂伊勢佐木店	045-261-1231
〃	有隣堂戸塚店	045-881-2661
〃	文華堂戸塚店	045-864-5151
〃	アーバン文華堂	045-821-5151
〃	文教堂青葉台南口店	045-983-5150
川崎市	有隣堂アゼリア店	044-245-1231
〃	有隣堂川崎BE店	044-200-6831
〃	文学堂本店	044-244-1251
〃	文教堂溝の口店	044-811-8258
鎌倉市	島森書店大船店	0467-46-3841
〃	鎌倉書店	0467-46-2619
横須賀市	平坂書房WALK店	0468-25-5537
藤沢市	有隣堂藤沢店	0466-26-1411
〃	リブロ藤沢店	0466-27-0111
〃	文教堂六会店	0466-82-9610
茅ヶ崎市	川上書店ルミネ店	0467-87-3827
平塚市	サクラ書店駅ビル店	0463-23-2751
〃	文教堂四之宮店	0463-54-2880
小田原市	八小堂書店	0465-22-7111
〃	伊勢治書店	0465-22-1366
〃	文教堂小田原店	0465-36-3677
厚木市	有隣堂厚木店	0462-23-4111
大和市	文教堂中央林間店	0462-75-4165
相模原市	文教堂相模大野店	0427-49-0650
〃	文教堂橋本店	0427-74-5581
相模原市	文教堂星ヶ丘店	0427-58-6121
津久井郡	文教堂城山店	0427-82-9278
〈東京〉		
千代田区	三省堂書店神田本店	03-233-3312
〃	書泉グランデ	03-295-0011
〃	東京堂書店	03-291-5181
〃	旭屋書店水道橋店	03-294-3781
〃	丸善お茶の水店	03-295-5581
〃	巌翠堂	03-291-1362
〃	いずみ神田南口店	03-254-8521
〃	明正堂秋葉原店	03-257-0758
〃	Bit INN 東京	03-255-4575
〃	T-ZONE	03-257-2660
〃	ラオックス THE COMPUTER館	03-5256-3111
中央区	八重洲ブックセンター	03-281-1811
〃	日本橋丸善	03-272-7211
〃	旭屋書店銀座店	03-573-4936
港区	書原新橋店	03-591-8738
〃	雄峰堂ＮＳ店	03-503-6586
〃	虎ノ門書房本店	03-502-3461
〃	虎ノ門書房田町店	03-454-2571
品川区	芳林堂大井町店	03-474-4946
〃	明屋書店五反田店	03-492-3881
渋谷区	紀伊國屋書店渋谷店	03-463-3241
〃	旭屋書店渋谷店	03-476-3971
〃	三省堂書店渋谷店	03-407-4545
〃	大盛堂書店	03-463-0511
〃	紀伊國屋書店笹塚店	03-485-0131
新宿区	紀伊國屋書店本店	03-354-0131
〃	三省堂書店新宿西口店	03-343-4871
〃	福家書店センタービル店	03-345-1246
〃	福家書店野村ビル店	03-342-0298
〃	新星堂ＮＳビル店	03-344-2055
〃	西武新宿ブックセンター	03-208-0380
〃	芳林堂高田馬場店	03-208-0241
〃	未来堂	03-200-9185
豊島区	旭屋書店池袋店	03-986-0311
〃	芳林堂池袋店	03-984-1101
〃	リブロ池袋店	03-981-0111
〃	三省堂書店池袋店	03-987-0511
〃	新栄堂本店	03-984-2345
〃	新栄堂アルパ店	03-988-0181
台東区	明正堂中通り店	03-831-0191
墨田区	ブックストア・談	03-635-1841
葛飾区	文教堂青戸店	03-838-5938
江戸川区	文教堂西葛西店	03-689-3621
大田区	アクトブックスサンカマタ店	03-735-1551
〃	竜文堂大森駅ビル店	03-775-3851
中野区	明屋書店東京本社	03-387-8451
杉並区	ブックセンター荻窪	03-393-5571
〃	書原杉並店	03-313-4778
武蔵野市	紀伊國屋書店吉祥寺東急店	0422-21-5543
〃	弘栄堂吉祥寺店	0422-22-1031
〃	パルコブックセンター吉祥寺	0422-21-8122
調布市	真光書店	0424-87-2222
府中市	啓文堂	0423-66-3151
三鷹市	三省堂書店三鷹店	0422-48-4510
〃	東西書房	0422-46-0275
小金井市	文教堂小金井店	0423-86-0161
国分寺市	三成堂国分寺店	0423-25-3211

特約書店基本図書一覧

定価はすべて税込です。

分類	書名	定価
アセンブラ	8086アセンブリ言語	●2,890円
	8086マクロプログラミング	●2,680円
	入門Turbo PASCAL Ver.5プログラミング	●3,300円
	GDCテクニカルブック	●3,500円
C言語	C言語の基礎知識	●2,580円
	C言語の活用理解	●2,060円
	C言語の応用50例	●2,370円
	上級・C言語の応用50例	●2,480円
	Play the C 上巻	●1,550円
	Play the C 下巻	●1,550円
	Cプリプロセッサ・パワー	●2,270円
	Turbo C 入門	●2,680円
	C++プログラミング	●2,680円
	Quick Cプログラミング	●2,680円
	詳説C言語	●4,500円
	MS-C Ver.5.1プログラミング	●3,300円
	Turbo C Ver.2.0プログラミング	●2,900円
機種別	ダイナブック・スーパーガイド	●3,200円
	最新ハードディスク入門	●2,600円
	最新EMS・RAMディスク入門	●2,500円
OS	プレイMS-DOS	●1,960円
	MS-DOSいたれりつくせり本	●1,860円
	新MS-DOS入門 ビギナー編	●1,900円
	新MS-DOS入門 シニア編	●2,300円
	新MS-DOS入門 応用編	●2,300円
	OS/2 APIブックI	●2,790円
	OS/2 APIブックII	●3,000円
	UNIXオペレーティング・ガイド	●3,090円
ワープロ	一太郎 Ver.3 ガイド	●2,580円
	入門一太郎 Ver.4.2	●2,500円
	P1 EXEガイド	●2,600円
ゲーム	RPG幻想事典	●1,550円
	RPG幻想事典 日本編	●1,860円
	魔法王国シムルグント	●1,860円
アプリケーション	LOTUS1-2-3ガイド ビギナー編	●2,480円
	LOTUS1-2-3 ガイドII	●2,580円
	桐Ver.2ガイド	●2,580円
	入門桐 Ver.2 一括処理	●3,500円
	Ninja3ガイド	●2,300円
	MS-Chart Ver.3.1ガイド	●2,990円
	まいとーくガイド	●2,370円
	dBASEIII PLUSガイド	●3,800円
	The CARD3ガイド	●2,900円
情報処理試験	アセンブラCASL入門	●2,060円
	ハードウェア徹底マスター	●2,580円
	FORTRAN徹底マスター	●2,890円
	受験用語ハンドブック	●1,860円
	情報処理入門 I 基礎知識	●1,240円
	情報処理入門 II 関連知識	●1,240円
	CASLで学ぶアセンブラ入門	●2,270円
	そっくり模擬試験	●2,200円

市町	書店名	電話番号
国立市	東西書店	0425-75-5061
小平市	文教堂小平店	0423-43-9229
東村山市	文教堂東村山店	0423-96-1115
立川市	オリオン書房ウイル店	0425-27-2311
八王子市	くまざわ書店本店	0426-25-1201
町田市	有隣堂町田店	0427-23-3018
〃	久美堂本店	0427-25-1330
〃	久美堂小田急店	0427-27-1111
〃	文教堂鶴川店	0427-35-4117
〃	文教堂小川店	0427-96-1781
多摩市	くまざわ書店桜ヶ丘店	0423-37-2531
福生市	文教堂福生店	0425-53-7708
〈甲信越・北陸〉		
甲府市	文教堂甲府店	0552-22-4600
長野市	平安堂長野店	0262-26-4545
〃	長谷川書店	0262-26-2122
松本市	ブックスロクサン	0263-35-5555
〃	改造社松本駅ビル店	0263-36-3777
〃	アクロスブックセンター	0263-32-5733
上田市	平安堂上田店	0268-22-4545
飯田市	平安堂飯田店	0265-24-4545
岡谷市	笠原書店	0266-23-5070
諏訪郡	平安堂下諏訪店	0266-28-1111
新潟市	紀伊國屋書店新潟店	025-241-5281
〃	萬松堂	025-229-2221
〃	北光社	025-228-2321
長岡市	覚張書店	0258-32-1139
〃	ブックセンター長岡	0258-36-1360
〃	長岡技大長峰文化	0258-46-6437
上越市	パソトピア コスモス	0255-25-5867
山北町	BOOKメディア	0254-77-3850
富山市	瀬川書店	0764-24-4566
〃	清明堂	0764-24-4166
〃	BOOKSなかだ豊田店	0764-32-1353
〃	文苑堂本郷店	0764-22-0552
〃	文苑堂赤江店	0764-33-0321
高岡市	文苑堂	0766-21-0333
〃	文苑堂横田店	0766-21-0431
金沢市	うつのみや片町店	0762-21-6136
〃	書林香林坊本店	0762-20-5011
野々市町	王様の本本店	0762-46-5325
福井市	勝木書店	0776-24-0428
〃	品川書店新田塚店	0776-24-1112
〈東海〉		
静岡市	静岡谷島屋呉服町本店	0542-54-1301
〃	江崎書店	0542-54-4481
〃	吉見書店	0542-52-0157
〃	戸田書店ＳＢＳ店	0542-81-5733
〃	戸田書店曲金店	0542-81-5899
沼津市	吉野屋	0559-23-5676
〃	マルサン書店宝塚店	0559-63-0350
富士市	戸田書店富士店	0545-51-5121
清水市	戸田書店本店	0543-65-2345
浜松市	浜松谷島屋連尺本店	0534-53-9121
名古屋市	三省堂書店名古屋店	052-562-0077
〃	星野書店近鉄ビル店	052-581-4796
〃	丸善名古屋支店	052-261-2251
〃	丸善ブックメイツセントラルパーク	052-971-1231
〃	日進堂上前津店	052-263-0550
名古屋市	三洋堂パソコンショップΣ	052-251-8334
〃	三洋堂いりなか本店	052-832-8202
〃	ちくさ正文館本店	052-741-1137
〃	白樺書房西店	052-774-7223
豊橋市	精文館	0532-54-2345
岡崎市	ブックス鎌倉	0564-54-1822
豊田市	三洋堂梅坪店	0565-35-2334
豊川市	三洋堂豊川店	05338-3-0334
刈谷市	三洋堂刈谷店	0566-24-1134
春日井市	三洋堂勝川店	0568-32-7806
一宮市	三洋堂一宮店	0586-77-5734
岐阜市	自由書房	0582-65-4301
大垣市	大洞堂ブックス258	0584-81-2553
〃	大洞堂岐大バイパス店	0584-74-7766
可児市	三洋堂可児店	0574-63-2334
多治見市	三洋堂多治見店	0572-24-0340
津市	別所書店11ビル店	0592-24-1014
四日市市	文化センター白揚	0593-51-0711
鈴鹿市	シェトワ白揚スズカ	0593-82-5221
〈近畿〉		
京都市	駸々堂京宝店	075-223-1003
〃	アバンティ・ブックセンター	075-682-5031
〃	オーム社書店河原町店	075-221-0280
〃	ジュンク堂京都店	075-252-0101
〃	オーム社書店竹田店	075-644-2611
奈良市	駸々堂大丸店	0742-26-6241
大阪市	旭屋書店本店	06-313-1191
〃	紀伊國屋書店梅田店	06-372-5821
〃	オーム社書店大阪店	06-345-0641
〃	駸々堂京橋店	06-353-3209
〃	駸々堂心斎橋店	06-251-0881
〃	旭屋書店ナンバ店	06-644-2551
〃	ナンバブックセンター	06-644-5501
〃	ヒバリヤ書店ナンバ店	06-644-5407
〃	旭屋書店アベノ店	06-631-6051
〃	ユーゴー書店	06-623-2341
〃	河村書店	06-951-2968
枚方市	水嶋書房京阪デパート店	0720-51-3432
高槻市	コーベブックス西武高槻店	0726-83-1766
東大阪市	ヒバリヤ書店本社	06-722-1121
神戸市	ジュンク堂センター街店	078-392-1001
〃	ジュンク堂サンパル店	078-252-0777
〃	海文堂書店	078-331-6501
〃	日東館書林	078-391-8701
姫路市	新興書房	0792-85-3344
〃	誠心堂書店	0792-81-2055
和歌山市	宮井平安堂	0734-31-1331
〃	帯伊書店	0734-22-0441
〈中国〉		
岡山市	紀伊國屋書店岡山店	0862-32-3411
〃	丸善岡山支店	0862-31-2261
津山市	津山ブックセンター	08682-6-4047
広島市	紀伊國屋書店広島店	082-225-3232
〃	丸善広島支店	082-247-2251
〃	金正堂	082-248-3715
〃	積善館	082-248-3151
尾道市	啓文社尾道店	0848-37-5151
福山市	啓文社福山店	0849-22-3111
福山市	ブックシティ啓文社	0849-25-0050
〃	啓文社コア	0849-41-0909
山口市	五十部誠文堂	0839-24-6630
〃	文栄堂	0839-22-5611
下関市	中野書店	0832-22-6181
宇部市	京屋書店	0836-31-2323
〃	末広書店	0836-31-0086
防府市	誠文堂国衙店	0835-25-1988
光市	三文字屋	0833-71-0251
鳥取市	富士書店	0857-23-7271
松江市	園山書店	0852-21-4167
〈四国〉		
徳島市	小山助学館本店	0886-54-2135
〃	小山助学館東口店	0886-25-1380
〃	森住丸善	0886-23-3228
高松市	宮脇書店本店	0878-51-3733
丸亀市	宮脇書店丸亀店	0877-22-5533
松山市	紀伊國屋書店松山店	0899-32-0005
〃	明屋書店本店	0899-41-4141
〃	明屋書店大街道店	0899-41-4242
〃	丸三書店	0899-31-8501
新居浜市	明屋星原店	0897-44-4000
宇和島市	明屋宇和島店	0895-23-1118
高知市	金高堂	0888-22-0161
〈九州・沖縄〉		
福岡市	紀伊國屋書店福岡店	092-721-7755
〃	りーぶる天神	092-713-1001
〃	積文館新天町店	092-781-2991
〃	福岡金文堂本店	092-741-2106
〃	福岡金文堂朝日ビル店	092-431-1094
〃	福岡金文堂デイトス店	092-451-6175
〃	福岡金文堂アニマート原	092-844-0088
北九州市	ナガリ書店	093-521-1044
〃	金栄堂	093-531-3685
〃	旭屋書店北九州店	093-631-6421
〃	井筒屋ブックセンター	093-641-0131
〃	カルパーク平野	093-661-7988
〃	白石書店本城店	093-601-2200
久留米市	エマックスたがみ	0942-33-1841
飯塚市	BOOKリード	0948-25-7266
大分市	パルコブックセンター大分店	0975-35-0643
〃	本町晃星堂	0975-33-0231
別府市	明林堂	0977-23-2183
宮崎市	中央、田中書店	0985-24-3511
〃	寿屋宮崎店	0985-27-4111
佐賀市	金華堂北バイパス店	0952-32-1965
〃	積文館佐賀店	0952-24-4314
〃	積文館デイトス店	0952-23-7155
長崎市	メトロ書店	0958-21-5453
〃	好文堂	0958-23-7171
佐世保市	金明堂書店	0956-22-4214
熊本市	紀伊國屋書店熊本店	096-322-5531
〃	長崎書店	096-353-0555
人吉市	明屋人吉店	0966-22-5486
鹿児島市	春苑堂ブックプラザ	0992-25-3200
〃	ブックスみすみ	0992-57-1011
那覇市	球陽堂書房ビル店	0988-63-3752
〃	文教図書	0988-62-1201

宇都宮店&大田原店

PRO SHOP

台数限定電子手帳特別セット

A SET
PA-8600
CE-20OL
CYBERNOTE PRO68k
Basic特価　¥41,800

B SET
PA-8600
CE-20OL
Stationary　PRO68k
Basic特価　¥37,800

PLAY THE MIDI MUSIC

A SET
CM-32L
SX-68M
MusicStudio Mu-1
Basic特価 ¥93,000

B SET
CM-64
SX-68M
MusicStudio Mu-1
Basic特価¥143,000

C SET
MT-32
SX-68M
MusicStudio Mu-1
Basic特価 ¥88,000

X68000用ハードディスク

アイテックITX-680……………………~~¥198,000~~
アイテックITX-640……………………~~¥158,000~~
アイテム　HXD-040……………………~~¥118,000~~
アイテム　HXD-042……………………~~¥128,000~~
ロジテックSHD-40……………………Basic特価
シャープ　CZ-64H……………………Basic特価

X68000用SCSI予約大募集

シャープ
光磁気ディスクユニット
CZ-6MO1　予約受付中
SCSIボード
CZ-6BS1　予約受付中

X68000用PRINTER'S

シャープCZ-8PC4……………………~~¥ 99,800~~
シャープCZ-8PG1……………………~~¥130,000~~
シャープCZ-8PG2……………………~~¥160,000~~
シャープCZ-PK10……………………~~¥ 99,800~~
エプソンAP-850……………………~~¥ 94,800~~
エプソンAP-550EX……………………~~¥ 62,800~~
エプソンVP-1350……………………~~¥ 94,800~~
エプソンVP-2050……………………Basic特価
NEC　PC-PR201GS……………………Basic特価
スターCR-3415CL……………………Basic特価

通信関連品

NEC COMSTER2424/4……………………Basic特価
NEC COMSTER2424/5……………………Basic特価
OMRON MD24FS5……………………~~¥ 49,800~~
OMRON MD24FS7……………………~~¥ 64,800~~
CommunicationProV2……………………~~¥ 19,800~~
た～みのる2……………………~~¥ 17,800~~

ゲーマー必須アイテム

CYBER STICK……………………~~¥ 23,800~~
XE1-AP……………………~~¥ 13,800~~
XE1-PRO……………………~~¥ 9,800~~
XE1-ST……………………~~¥ 4,900~~

グラフィックツール

スキャナパラレルボード……………………~~¥ 29,800~~
CZ-8NS1……………………~~¥ 19,800~~
GT-6000……………………Basic特価
GT-1000……………………~~¥ 79,800~~
HS-10RII……………………~~¥ 49,800~~
HS-7RII……………………~~¥ 39,800~~
CZ-6VP1……………………~~¥198,000~~
IO-735X……………………~~¥248,000~~
ジーズスタッフPRO68K……………………Basic特価
デジタルクラフト……………………~~¥ 39,800~~
マジックパレット……………………Basic特価
サイクロンExpress α68……………………~~¥ 98,000~~

その他周辺機器

拡張I/O BOX……………………~~¥ 88,000~~
アンプ内蔵スピーカー……………………~~¥ 36,600~~
カラーイメージユニット……………………~~¥ 69,800~~
ビデオボード……………………~~¥ 21,000~~
RGBシステムチューナー……………………~~¥ 33,100~~
CRTフィルター……………………~~¥ 19,800~~

CRT

CU-21HD……………………~~¥148,000~~
CZ-605D……………………~~¥115,000~~
CZ-604D……………………~~¥ 94,800~~
CZ-613D……………………~~¥138,000~~
CZ-603D……………………~~¥ 84,800~~

NEWS 1

ロケットキャッシャー完成!
新アルゴリズムの採用により従来比約3倍の高速化を実現。
ハードディスクキャッシャーのみのバージョンアップとなります。旧製品のディスクのラベルを同封のうえ、1,500円(送料・税込)を現金書留でお送り下さい。

NEWS 2

ビデオボード(CZ-6BV1)を外付けに!
ビデオボード収納ケース(KGB-BVBX)
近日発売予定(通信販売のみで一般販売はしません)

全国どこでも発送可　長期クレジットOK　送料全国均一¥1,000　宅配便にて即日配送

株式会社計測技研
本社営業部/マイコンショップ/通販部　宇都宮市竹林町503—1　TEL0286-22-9811　FAX0286-25-3970
大田原営業所/マイコンショップ　大田原市美原1—13—4　TEL0287-23-5352　FAX0286-23-5364

マイコンショップ BASIC HOUSE お申し込み・お問い合せは ☎0286-22-9811(代)

オーエーブレイン　全国通販

●オフコンからパソコンまで
幅広〜い品揃え。おまかせあれ!! お電話くださいネ!

☎03-5688-3621

★全商品保証書付。専門のアドバイザーがお客様のニーズに親切に対応します。
★初期不良・輸送トラブル等に迅速に対応し、即交換させていただきます。
★送料は1個につき¥1,000です。(※一部離島は除きます。お問合せ下さい。)

●ご注文、お問合せは…毎日午前10時から午後8時まで
●下取・買取は電話で見積りしております。責任を持って下取りさせて頂きます
●商品のお届けは…入金確認後、即日発送致します。

OAB特選〜X68000シリーズセット (ゲームパック・ディスケット付) (税抜き)

①X68000 EXPERTII
●CZ-603C-BK/GY
●CZ-605D-BK/GY
●MD-2HD 20枚
定価合計¥453,000
●SX-WINDOW搭載!!

OAB大特価

②X68000 EXPERTII-HD
●CZ-613C-BK/GY
●CZ-605D-BK/GY
●MD-2HD 20枚
定価合計¥563,000

OAB大特価

③X68000 PROII
●SX-WINDOW搭載!!

●CZ-653C-BK/GY
●CZ-605D-BK/GY
●MD-2HD 20枚
定価合計¥400,000

OAB大特価

④X68000 PROII-HD
●CZ-663C-BK/GY
●CZ-605D-BK/GY
●MD-2HD 20枚
定価合計¥510,000

OAB大特価

X68000 SUPER-HD ●SX-WINDOW搭載!!
●SX-WINDOW搭載
●SCSIインターフェース装備
●80MBハードディスク搭載
●3MB大容量メモリ装備
●高解像度グラフィック

⑤X68000SUPER-HD
●CZ-623C-TN(チタン)
●CZ-613D-TN(チタン)
●MD-2HD 20枚
定価合計¥633,000

OAB大特価

X68000 EXPERT-HD=特選限定品

X68000 EXPERT-HD
●CZ-612(BK)
(定価¥466,000)
●CZ-605(BK)
(定価¥115,000)
定価合計¥581,000

OAB特価¥368,000

特選CRT
●CZ-605D
(¥115,000)………大特価!!
●CZ-613D
(¥135,000)………大特価!!
●CZ-604D
(¥ 94,800)………大特価!!
●CU-21HD
(¥148,000)………大特価!!

周辺機器コーナー

プリンターセットコーナー
●CZ-6PVI(カラービデオプリンター)
定価¥198,000…………▶特価¥152,000
●CZ-8PC3(24ドット熱転写カラープリンター)
定価¥ 65,800…………▶特価¥ 53,000
●CZ-8PK10(24ピン漢字ドットプリンター・136桁)
定価¥ 97,800…………▶特価¥ 73,000
●CZ-8PGI(24ピンカラー漢字ドットプリンター・80桁)
定価¥130,000…………▶特価¥ 98,000
●CZ-8PG2(24ピンカラー漢字ドットプリンター・136桁)
定価¥160,000…………▶特価¥119,000
●IO-735X(カラーイメージェットプリンター)
定価¥248,000…………▶特価¥185,000

■CZ-8PC4(定価¥99,800)
特選品!!

●48ドット熱転写カラー漢字プリンター

特価~~¥64,800~~

X68000用ソフトウェアー・コーナー
①CZ-212BS(BUSINESS)…………定価¥ 68,000▶特価¥ 53,000
②CZ-220BS(DATA)…………定価¥ 58,000▶特価¥ 45,000
③CZ-215MS(Sampling)…………定価¥ 17,800▶特価¥ 13,800
④CZ-221HS(NEW Print Shop)……定価¥ 10,800▶特価¥ 15,500
⑤CZ-227BS(TOP財務会計)………定価¥200,000▶特価¥158,000
⑥CZ-226BS(CARD)…………定価¥229,800▶特価¥ 23,000
⑦CZ-223CS(Communicabion)………定価¥ 19,800▶特価¥115,500
⑧CZ-213MS(MUSIC)…………定価¥ 18,800▶特価¥ 14,800
⑨CZ-211LS(C compiler)…………定価¥ 39,800▶特価¥ 31,000
⑩C-TRACE(キャスト)…………定価¥ 68,000▶特価¥ 52,000
⑪EW(イースト)…………定価¥ 38,000▶特価¥ 29,000

X68000用周辺機器コーナー
●CZ-6BE1B…… 定価¥ 28,000▶特価¥ 22,000
●CZ-6BM1…… 定価¥ 26,800▶特価¥ 21,000
●CZ-6EB1…… 定価¥ 88,000▶特価¥ 69,800
●CZ-6VT1…… 定価¥ 69,800▶TEL下さい
●CZ-8NS1…… 定価¥188,000▶特価¥149,000
●CZ-6BC1…… 定供¥ 79,800▶特価¥ 63,000

今月の特価品(限定)お早目に!!

★CZ-652C(BK)+CZ-602D(BK)
4セット限り …… 大特価¥258,000

●SHARP WD-A300(ワープロ)
定価¥165,000…………特価¥110,000
●SHARP WD-A330(ワープロ)
定価¥185,000…………特価¥125,000
●SHARP WD+HL30(ワープロ)
定価¥198,000…………特価¥134,000
●SHARP PW-910(ワープロ)
…………特価¥ 85,000
●NEC PC-KD853(アナログCRT)
…………特価¥ 50,000
●三菱XC-1498C(アナログCRT)
…………特価¥ 54,800
●SHARP CU-14FD(アナログCRT)
…………特価¥ 46,000
●SHARP PA-8500(電子手帳)
…………特価¥ 16,000

通信販売によるご購入方法(お電話でお申し込み下さい。)

現金一括払い
銀行振込: 電信扱いにてお振込下さい
手数料はお客様負担となります
現金書留: 住所、氏名、電話番号、商品名、使用機種、メディア等をお書き添えのうえ、現金書留にて当社までお送り下さい

クレジット
専用のお申し込み用紙をお送り致しますので、必要事項をご記入・捺印のうえ、ご返送下さい
※未成年者の方は、保護者のご承認を受けてからお申し込み下さい

振込先
●第一勧業銀行　御徒町支店
(普)1376679　オーエーブレイン
●朝日信用金庫　本店
(普)334833　オーエーブレイン

★クレジットは1〜60回払いで月々5,000円よりご自由に設定できます。

オーエーブレイン　〒110 東京都台東区台東1-28-4
TEL & FAX 5688-3621

I・O DATA 増設RAMボード

●1MB増設PAMボード
PIO-6BEI-A
定価
¥25,000

特価¥18,800

●2MB増設RAMボード
PIO-6BE2-2M
定価
¥50,000

特価¥37,800

●4MB増設RAMボード
PIO-6BE4-4M
定価
¥88,000

特価¥65,800

■ハードディスク　■特価品もありますのでTEL下さい。

●アイテック ITX-640…………特価¥117,000
●アイテック ITX-680…………特価¥149,000
●ロジテック LHD-32V…………特価¥ 85,000
●ロジテック LHD-34VE…………特価¥ 90,000
●ロジテック LHD-34V…………特価¥104,000
●シャープ CZ-620H…………特価¥118,000
●シャープ CZ-64H…………特価¥ 95,000
●アイテム HXD-040…………特価¥ 88,000
●アイテム HXD-042…………特価¥ 95,000
●ICM SR 80…………特価¥130,000

中古パソコン (価格/在庫は変動します。予約は5日以内とします。)

PC-9801RA5…………¥338,000より
PC-9801RA2…………¥265,000より
PC-9801RX2…………¥199,000より
PC-9801EX2…………¥190,000より
PC-9801VX21…………¥170,000より
PC-9801UX21…………¥165,000より
PC-9801VX2…………¥160,000より
PC-9801VM21…………¥150,000より
PC-9801UV11…………¥148,000より
PC-9801LV22…………¥160,000より
PC-286VE…………¥150,000より
PC-286US…………¥155,000より
PC-286VS…………¥165,000より
CZ-600C…………¥160,000より
CZ-601C…………¥170,000より
CZ-611C…………¥198,000より
CZ-652C…………¥178,000より
CZ-612C…………¥210,000より
68000用モニター…………¥ 49,000より
PC-9801用サウンドボード…………¥ 13,000より
PC-88SR, FR…………¥ 50,000より
PC-88FH, FA…………¥ 65,000より
400ラインCRT…………¥ 38,000より
200ラインCRT…………¥ 10,000より

オーエーブレイン今月の特価品!! 台数限定 お早目に!!

ドライブ・ユニット
アクセル
●FDC-357……特価¥36,000
●FDC-358……特価¥49,000
コンピュータ・リサーチ
●CRC-FD3.5S…特価¥29,000
●CRC-FD3.5W…特価¥42,000
グローリア
●GD-35M1……特価¥23,000
●GD-35M2……特価¥39,000
緑電子
●Little-F1……特価¥26,000
●Little-F2……特価¥38,000
SNE
●SNE-2……特価¥49,000

プリンター
NEC
●PC-PR201G+
PC-PR201G-04……特価¥ 99,800
●NM-4150……特価¥132,000
エプソン
●AP-850PC……特価¥ 64,000
●VP-2050PC……特価¥ 99,000
キャノン
●BJ-130J……特価¥125,000

ハード・デイスク
■ARK WOOD
NEC純正ドライブ使
●AW-N40C
(定価¥138
…特価¥ 94
●AW-N100C
(定価¥195
…特価¥134

サウンド・ボード
SNE
1 サウンドオーケストラV……特価¥23,000
2 サウンドオーケストラL……特価¥17,800
3 リトルオーケストラ……特価¥14,500
4 リトルオーケストラL……特価¥12,000
5 サウンド・ミュージシャン…特価¥16,000
6 リトルミュージシャン……特価¥ 8,500

ワープロ
NEC
●PWP-70HR……特価¥175,
●PWP-70R……特価¥124,
●PWP-50RD……特価¥119,
東芝……特価¥140,
●JW-95HD……特価¥140,
●JW-95F……特価¥108,
キャノン
●CW-α350……特価¥155,
その他、シャープ、松下、富士通等、TEL

■流通事情により、広告表示よりお安くなる場合もございます。まずは、お電話下さい。■ビジネス・ゲームセットもございます。

株式会社 デンキヤ

営業時間AM11:00〜PM7:00 水・木曜定休

（価格は全べて税込みです）

セット超特価 X68000 PERSONAL WORKSTATION PRO II・PRO II HD

CZ-653C
CZ-604D
セット¥特価
¥24,000×12回
¥12,700×24回

CZ-653C
CZ-605D
セット¥特価
¥25,300×12回
¥13,400×24回

CZ-603C
CZ-604D
セット¥特価
¥27,300×12回
¥14,500×24回

CZ-603C
CZ-605D
セット¥特価
¥28,600×12回
¥15,100×24回

セット超特価 X68000 PERSONAL WORKSTATION EXPERT II・EXPERT II HD

CZ-663C
CZ-605D
セット¥特価
¥32,200×12回
¥17,000×24回

CZ-663C
CZ-613D
セット¥特価
¥33,500×12回
¥17,700×24回

CZ-613C
CZ-613D
セット¥特価
¥36,800×12回
¥19,500×24回

CZ-623C
CZ-613D
セット¥特価
¥39,900×12回
¥21,200×24回

全品メーカー保証 即決クレジットOK

ディスプレイ		プリンタ		周辺機器		ソフト	
CZ-604D	特価	CZ-8PC4	特価	CZ-8NJ1	¥1,400	CZ-213MS	¥15,500
CZ-605D	特価	CZ-8PG1	特価	CZ-8NJ2	¥18,540	CZ-223CS	¥15,300
CZ-613D	特価	CZ-8PG2	特価	PIO-6BE1A	¥20,000	CZ-219SS	¥23,100
CU-21HD	特価	IO-735X	特価	PIO-6BE2	¥39,000	CZ-211LS	¥30,800

24時間テレホンサービス
0482-54-3444

お申し込み
TEL.0482-54-3400
FAX.0482-54-3443
埼玉県川口市西川口4-6-4

お支払い
下記取引銀行口座
までお振込み下さい。
三菱銀行西川口支店
㈱デンキヤ㊟0258081

Sofmap 商品は今すぐ！お支払いは9ヶ月後から。

For Computer Communication Age

株式会社ソフマップ　冬のボーナス一括払い、冬・夏2回払いも受付中!!

●この表の価格は7月23日現在のものです。●掲載価格には消費税が含まれておりません。

基本セット SUPER-HD

大容量80MB、3.5インチHD内蔵、SCSIインターフェイス標準装備、SX-WINDOW搭載

月々¥3,000から

クレジット注文 No.1

CZ-623C-TN〈本体〉	¥Sofmap特価
CZ-613D-TN〈15″ドットピッチ0.31〉	¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉	¥Sofmap特価
標準価格¥650,000	¥お電話にて

¥3,000×84回	ボーナス ¥30,000×12回
¥5,400×36回	ボーナス ¥60,000×6回
¥8,000×84回	ボーナス なし
¥10,100×60回	ボーナス なし
¥12,000×48回	ボーナス なし

ワープロセット SUPER-HD

月々¥3,700から

クレジット注文 No.2

CZ-623C-TN〈本体〉	¥Sofmap特価
CZ-613D-TN〈15″ドットピッチ0.31〉	¥Sofmap特価
CZ-8PC4〈48ドット熱転写プリンタ〉	¥Sofmap特価
Hyper Word〈CZ-251BS日本語ワープロ〉	¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉	¥Sofmap特価
標準価格¥789,600	¥お電話にて

¥3,700×72回	ボーナス ¥43,000×12回
¥7,200×36回	ボーナス ¥70,000×6回
¥9,800×84回	ボーナス なし
¥12,300×60回	ボーナス なし
¥14,700×48回	ボーナス なし

ゲーム＆通信セット SUPER-HD

月々¥3,400から

クレジット注文 No.3

CZ-623C-TN〈本体〉	¥Sofmap特価
CZ-613D-TN〈15″ドットピッチ0.31〉	¥Sofmap特価
CZ-8PC4〈48熱転写プリンタ〉	¥Sofmap特価
CZ-8NJ2〈アナログスティック〉	¥Sofmap特価
ゲームソフト2本〈定価¥9,800以下のお好きなソフト〉	¥Sofmap特価
MD-24FS5〈通信用モデム2400BPS〉	¥Sofmap特価
CZ-257CS〈COMMUNICATION PRO68K Ver.2.0〉	¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉	¥Sofmap特価
標準価格¥862,800	¥お電話にて

¥3,400×60回	ボーナス ¥60,000×10回
¥7,200×36回	ボーナス ¥80,000×6回
¥10,700×84回	ボーナス なし
¥16,000×48回	ボーナス なし
¥20,400×36回	ボーナス なし

基本セット EXPERTII-HD

集積度を高めた"マンハッタンシェイプ"3Mの大容量メモリ、SX-WINDOW搭載

月々¥2,500から

クレジット注文 No.4

CZ-613C-BK〈本体〉	¥Sofmap特価
CZ-613D-BK〈15″ドットピッチ0.31〉	¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉	¥Sofmap特価
標準価格¥600,000	¥お電話にて

MIDIセット EXPERTII-HD

月々¥3,000から

クレジット注文 No.5

CZ-613C-BK〈本体〉	¥Sofmap特価
CZ-605D-BK〈15″ドットピッチ0.39〉	¥Sofmap特価
CZ-6BM1〈MIDIボード〉	¥Sofmap特価
CZ-247MK〈MUSIC PRO-68K〉	¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉	¥Sofmap特価
標準価格¥635,600	¥お電話にて

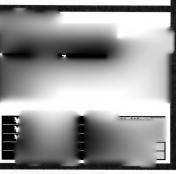
¥3,000×84回	ボーナス ¥30,000×14回
¥8,900×72回	ボーナス なし
¥10,300×36回	ボーナス ¥30,000×6回
¥15,300×36回	ボーナス なし
¥22,200×24回	ボーナス なし

データベースセット EXPERTII-HD

月々¥3,600から

クレジット注文 No.6

CZ-613C-BK〈本体〉	¥Sofmap特価
CZ-613D-BK〈15″ドットピッチ0.31〉	¥Sofmap特価
CZ-226BS〈CARD PRO68K〉	¥Sofmap特価
CZ-8PK10〈24ピンプリンター130桁〉	¥Sofmap特価
標準価格¥727,600	¥お電話にて

¥3,600×84回	ボーナス ¥33,000×14回
¥5,600×60回	ボーナス ¥35,000×10回
¥7,600×36回	ボーナス ¥60,000×6回
¥9,100×84回	ボーナス なし
¥11,500×60回	ボーナス なし

ゲームセット EXPERTII

月々¥2,400から

クレジット注文 No.8

CZ-603C〈本体〉	¥Sofmap特価
CZ-605D〈15″ドットピッチ0.39〉	¥Sofmap特価
CZ-8NJ2〈アナログスティック〉	¥Sofmap特価
ゲームソフト2本〈定価¥9,800以下のお好きなソフト〉	¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉	¥Sofmap特価
標準価格¥513,400	¥お電話にて

[illegible]	ボーナス ¥28,000×12回
[illegible]	ボーナス ¥48,000×6回
[illegible]	ボーナス なし
[illegible]	ボーナス なし
[illegible]	ボーナス なし

グラフィックセット EXPERTII

月々¥2,200から

クレジット注文 No.9

CZ-603C〈本体〉	¥Sofmap特価
CZ-605D〈15″ドットピッチ0.39〉	¥Sofmap特価
Z'S STAFF PRO68K Ver.2.0〈グラフィックツールソフト〉	¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉	¥Sofmap特価
標準価格¥528,000	¥お電話にて

¥2,200×60回	ボーナス ¥35,000×10回
¥4,000×36回	ボーナス ¥50,000×6回
¥7,100×72回	ボーナス なし
¥9,600×48回	ボーナス なし
¥12,300×36回	ボーナス なし

ビジネスセット EXPERTII

月々¥2,300から

クレジット注文 No.10

CZ-603C〈本体〉	¥Sofmap特価
CZ-605D〈15″ドットピッチ0.39〉	¥Sofmap特価
CZ-8PC4〈48ドット熱転写プリンタ〉	¥Sofmap特価
CZ-212BS〈BUSINESS PRO68K〉	¥Sofmap特価
マクセルブランクディスケット〈5″2HD×10枚〉	¥Sofmap特価
標準価格¥637,800	¥お電話にて

¥2,300×60回	ボーナス ¥46,000×10回
¥4,000×36回	ボーナス ¥68,000×6回
¥8,000×84回	ボーナス なし
¥12,000×48回	ボーナス なし
¥15,300×36回	ボーナス なし

CZ-8PC4 定価¥99,800 ¥お電話にて

CZ-8PK10 定価¥97,800 ¥お電話にて

CZ-8PG1 定価¥130,000 ¥お電話にて

CZ-8PG2 定価¥160,000 ¥お電話にて

周辺機器

	定価	ソフマップ特価
●CZ-6BE1	¥35,000	¥26,500
●CZ-6BE1A	¥38,000	¥28,600
●CZ-6BE2	¥79,800	¥60,000
●CZ-6BE4	¥138,000	¥107,000
●CZ-6BF1	¥49,800	¥38,200
●CZ-6BP1	¥79,800	¥61,000
●CZ-6TU	¥33,100	¥25,000
●AN-S1000	¥36,600	¥28,500
●CZ-8NS1	¥188,000	¥145,000
●CZ-6EB1	¥88,000	¥67,500

SOFT WARE

	定価	ソフマップ特価
●Z's STAFF PRO68K V2.0	¥58,000	¥39,700
●DATA PRO68K(CZ-220BS)	¥58,000	¥お電話にて
●CARD PRO68K(CZ-226BS)	¥29,800	¥お電話にて
●Cコンパイラ PRO68K V2.0(CZ-245LS)	¥39,800	¥32,000
●SOUND PRO68K(CZ-214MS)	¥15,800	¥12,500
●MUSIC PRO68K(CZ-213MS)	¥15,800	¥お電話にて
●サンプリング PRO68K(CZ-215MS)	¥17,800	¥14,000
●コミュニケイション V2.0(CZ-257CS)	¥19,800	¥お電話にて
●OS-9(CZ-219SS)	¥29,800	¥お電話にて
●KAMIKAZE	¥68,800	¥46,000

No.1システム

No.1 下取りシステム

お持ちの機種を下取りに出して、新品に買替えようと思っている方、ソフマップに御相談下さい。
買取り価格がどこよりも高く、新品の販売価格がどこよりも安いから、当然どこよりもお得な条件でお買求めいただけます。
又、差額を商品券でお支払いもできます。

No.1 配送システム

1.到着日指定、夜間配送システム
お客様のご都合に合わせて配送させていただきます。機種によっては、夜間配送できないものがあります。

2.代金引換システム（要手数料）
係員が品物をお届けに行きますので、その時にお支払い下さい。

No.1 クレジットシステム

1. 9ヶ月先からのお支払いOK
スキップクレジットを御利用になれば支払い開始月を1ヶ月から、最長9ヶ月先までおくらせる事が出来ます。

2. 月々¥1,000からのお支払いOK
月々のお支払い金額の設定が¥1,000からOK。

3. 84回払いもOK
お客様の予算に合わせて、1回から最長84回まで支払い回数をお選びいただけます。

4. ステップアップクレジット
お客様のプランに合わせて、毎月のお支払い金額を徐々に増やしていくシステムです。例えば、1年目は¥3,000、2年目は¥6,000というように、御自由に設定することができます。

5. ボーナス8回払いもOK
毎月の支払いは0(ゼロ)、ボーナス時のみのお支払いでクレジットが御利用になれます。回数は1、2回の他、4・6・8回払いまでOK。

6. カードクレジット
各種クレジットカードが店頭だけでなく、通信販売でも御利用になれます。詳しくはお気軽にお問い合わせ下さい。

7. カレッジクレジット
保証人なしで、学生の方でもクレジットが御利用できます。(20歳以上)

No.1 サポートシステム

1. 初期不良交換期間3ヶ月
●万一、お届けした商品が不良の場合、お買い上げ日より3ヶ月以内なら、同等品と即、交換致します。

2. 新品パソコン3年保証
●メーカー保証が1年の場合、メーカー保証1年＋マップ保証2年の計3年間の保証になります。

3. 中古パソコン1年保証
●中古パソコン本体は、1年間保証致します。（ディスプレイプリンタ等は6ヶ月保証となります）

4. 新品パソコン買取り保証
●1ヶ月以内であれば必ず買取り保証金額で、下取り、買取り致します。

5. 永久買取り保証
●古くなったパソコン、スクラップ寸前のパソコンでもOK!!
どんなパソコンでも、どこよりも高く買い取ります。

注文書

ソフト名	機種	メディア	販売価格

同人ソフト募集中！

お名前

住所

TEL

買取依頼書

ソフト名	機種	メディア	販売価格

「自宅で学ぶアセンブラ」通信教育受付中

全国通販

中古ソフト高価買取中！

新品ソフト15%OFF 送料、消費税込み。

ただし、北海道・沖縄、離島の方は200円プラスして送金して下さい。
定価5,000円未満の商品についてはプラス300円。

PC98シリーズ

商品名	定価(円)	販売価格
三国志II	14,800	12,500
ポピュラス	9,800	8,300
プロミストランド	4,800	4,300
ダンジョンマスター	9,800	8,300
シムシティー(9月7日発売)	9,800	8,300
大航海時代	9,800	8,300
キャンペーン版大戦略II	9,800	8,300
栄冠は君に	9,500	8,000
FOXY	6,800	5,700
ドラゴンナイト	6,800	5,700
D-欧州蜃気楼	12,800	10,800
パズルトピア	7,800	6,600
エイトレイクスゴルフクラブ	4,800	4,300
雀皇登竜門	9,800	8,300
ドラゴンスレイヤーVI	8,700	7,300
バトル	12,800	10,800
46億年物語	9,800	8,300
機甲師団	9,500	8,000
天と地と	12,800	10,800
RYU	11,600	9,800
ロンメル	8,800	7,400
戦略空軍	8,800	7,400
エメラルドドラゴン	9,800	8,300
ストロベリー大戦略	6,800	5,700
デ・ジャ	6,800	5,700
陽炎迷宮	8,800	7,400
DUEL	8,700	7,300
インペリアルフォース	8,800	7,400
大戦略III「赤の逆襲編」	3,600	3,300
プリンスオブペルシャ	8,800	7,400
キャンペーン版大戦略IIマップ	4,800	4,000
麻雀悟空ー天竺への道	9,800	8,300
クォータースタッフ	9,800	8,300
サイレントメビウス	14,800	12,500

3.5"版も在庫あります。

PC88シリーズ

商品名	定価(円)	販売価格
DUEL	8,700	7,300
三国志II	14,800	12,500
FOXY	6,800	5,700
ドラゴンナイト	6,800	5,700
雀ボーグすずめ	7,800	6,600
大航海時代	9,800	8,300
天使たちの午後番外3	8,800	7,400
セーラー服戦士フェリス	6,800	5,700
エメラルドドラゴン	8,800	7,400
クリムゾンIII	8,700	7,300
夢幻の心臓III	9,700	8,200
鳴門巻秘帖	6,800	5,700
きゃんきゃんバニースペリオール	6,800	5,700
リップスティックADV II	6,800	5,700
DPS	5,400	4,500
トンネルズ&トロールズ	9,800	8,300
うろつき童子	6,800	5,700
ランペルール	9,800	8,300
ランペルールCD付	12,200	10,300

その他多数在庫あり

X68000シリーズ

商品名	定価(円)	販売価格
グラナダ	8,800	7,400
天下統一	9,800	8,300
サーク	8,800	7,400
パズニック	7,800	6,600
トンネルズ&トロールズ	9,800	8,300
ダンジョンマスター	9,800	8,300
ポピュラス	9,800	8,300
ワンダラーズフロムイース	8,700	7,300
ナイトアームス	9,700	8,200
バブルボブル	7,200	6,100
ダウンタウン熱血物語	8,800	7,400
スーパーハングオン	8,800	7,400
三国志II	14,800	12,500
闇の血族(上巻)	8,800	7,400
ワールドコート	8,800	7,400
ルーンワース	8,800	7,400
シムシティー(9月7日発売)	9,800	8,300
クォース	6,800	5,700
ガンシップ	11,800	10,000
提督の決断	14,800	12,500
イースI	9,800	8,300

その他多数在庫あり

中古ソフト大処分

販売価格：☎にてお問い合わせください。

PC98シリーズ

商品名	定価(円)	販売価格
三国志II	14,800	
ポピュラス	9,800	
ダンジョンマスター	9,800	
ブライ上巻	9,800	
キャンペーン版大戦略2	9,800	
トンネルズ&トロールズ	9,800	
FOXY	6,800	
ドラゴンナイト	6,800	
セーラー服戦士フェリス	6,800	
うろつき童子	6,800	
ダークレイス	9,600	
エイトレイクスゴルフクラブ	4,800	
レナム	9,800	
雀皇登竜門	9,800	
ドラゴンスレイヤーVI	8,700	
維新の嵐	9,800	
提督の決断	14,800	
水滸伝	9,800	
バトル	12,800	
ワンダラーズフロムイース	8,700	
46億年物語	9,800	
機甲師団	9,500	
戦略空軍	8,800	
ロンメル	8,800	
天と地と	12,800	
RYU	11,600	
ロードス島戦記	9,800	
ブルトンレイ	8,800	
エメラルドドラゴン	9,800	
デジャ	6,800	
斬アナログ	9,800	
アークス2	9,800	

3.5"版も在庫あります。

PC88シリーズ

商品名	定価(円)	販売価格
ドラゴンナイト	6,800	
FOXY	6,800	
DUEL	8,700	
ドラゴンスレイヤーVI	8,700	
信長戦国群雄伝	9,800	
水滸伝	9,800	
三国志II	14,800	
銀河英雄伝説	8,800	
大航海時代	9,800	
サバッシュ	7,800	
トンネルズ&トロールズ	9,800	
雀ボーグすずめ	7,800	
ソーサリアン	9,800	
イース1	7,800	
イース2	7,800	
イース3	8,700	
夢幻の心臓III	9,700	
きゃんきゃんバニースペリオール	6,800	
ストロベリー大戦略	6,800	
DPS	5,400	
維新の嵐	9,800	
アークス2	9,800	
ラストハルマゲドン	7,800	
ルーンワース	8,800	

その他多数在庫あり

X68000シリーズ

商品名	定価(円)	販売価格
アーススス2	9,800	
アールタイプ	7,800	
アフターバーナー	9,200	
イース3	8,700	
維新の嵐	9,800	
信長戦国群雄伝	9,800	
ガウディー	9,800	
グラナダ	8,800	
エージャックス	8,800	
ジェノサイド	8,800	
ナイトアームス	9,700	
サラマンダー	8,800	
スーパーハングオン	8,800	
天下統一	9,800	
ダンジョンマスター	9,800	
ポピュラス	9,800	
デスブリンガー	9,800	
大海令	12,800	
ラストハルマゲドン	9,800	
森田将棋2	10,000	
メタルサイト	8,800	
V'BALL	7,900	
源平闘魔伝	7,800	

その他多数在庫あり

新作エメラルド伝説¥6,600 送料・消費税要りません！

新作DE・JA¥5,700 送料・消費税要りません！

- 代金は注文書を添えて、現金書留で送って下さい。(小為替不可)
  後払いシステムもあります。
- 新品ソフトをご注文の場合は、商品代金を送って下さい。(送料、消費税込み)
- 中古ソフトをご注文の場合は、必ず電話にて在庫確認をして下さい
- 未発売ソフトの場合は、予約扱いとさせていただきます。

- 買取り希望の場合は、まずソフトを当店に送って下さい。こちらで高額査定のうえ、TELでご連絡させていただきます。値段が合わない場合、商品はすぐ返送しますので、安心してお送り下さい。
- ディスケットの送料は、100枚まで500円です。
- 中古ソフトリスト完成、ご希望の方は62円切手3枚をお送り下さい。

ブランド品	5"2HD	10枚	1,000円
ノーブランド	5"2HD	10枚	600円
ノーブランド	5"2D	10枚	400円
ブランド品	3.5"2HD	10枚	2,800円
ノーブランド	3.5"2HD	10枚	1,500円

消費税3%及び送料500円をプラスして送金して下さい。

DISKシャトル高槻

〒569
大阪府高槻市高槻町12-13
明和ビル2F

TEL受付時間 AM11:00~PM8:00
営業時間 AM12:00~PM8:00

<東京地区TELオープン>

☎0726-83-9907 ☎03-713-1424

アイ・ツー EXE CLUB

新規ユーザー・EXE会員大集合

★X68000ユーザーニーズに対応したハード・ソフトウエア・周辺機器は全て展示しています。
★新製品情報・ユーザー同士の情報交換ができる、メンバー様の憩いのスペースです。
★大特価セール期間中X68000・ディスプレイ・プリンター御購入の方は全国どこでも送料無料!!
★遠くでなかなかお越し頂けない方にも通販専用TELで専門スタッフ(X68 PRO STAFF)が親切丁寧にお答えします。
★X68000お買い上げの方、アイツーよりBigプレゼント。

X68000 オリジナルステッカー
X68000 フロッピータイトルシール
X68000 オリジナルテレフォンカード
X68000 バッグ
} お好きなもの2点もれなくついてくる!!

★現在シャープX68000 EXE会員の方、おトモダチをご紹介下さい。ご購入成立時点でアイ・ツーとシャープよりステキなプレゼント進呈中!!
★アイ・ツーメンバーズ優待制度実施
アイ・ツーでX68000・及びソフトウエア周辺機器をお買上げ頂きましたユーザー様にはオリジナルメンバーズカードを送付致します。メンバーズの方には楽しいパソコンライフをおくれますように最善のフォローをアイ・ツーより提供します。

新製品入替機 大放出!!

展示品処分祭 早い者勝ち

EXPERT SET CZ-612CBK CZ-603DBK **40% off**

定価+3%=¥567,324

↓

あなたのお支払はタッタの **¥338,000** ポ・ッ・キ・リ

シャープ販売コンテスト・パソコン部門最高峰賞
シャープ販売第一位受賞感謝セール!
■期間/8月18日〜9月17日

アイ・ツーinシャープグランドフェア'90 OSAKAスタジアムに多数のご来場頂きまして、誠にありがとうございました。アイ・ツーサンクスフェアPart2も只今企画中ですので、乞うご期待!!

X68000プロショップ(専門店)ならではの企画です!!

SX-WINDOW ってなぁ〜に?

X68000ユーザーみんな集まれ! SX-WINDOWの勉強会?を開催しまーす。

参加ご希望の方は、62円切手同封のうえ、お名前・ご住所・TEL・生年月日・お持ちのX68000の型番を書いて、アイ・ツーEXE CLUBあてで、おくって下さい。
日時、場所etc...ご連絡します!!

場所はとりあえず大阪です!

X68000ユーザーとっておきのグッズ!!
X68000ユーザーのステータスシンボル。
新グッズもグループインしてますます充実。
キミのパソコンライフが一層楽しくなるコレクションだ!
X68000オリジナルグッズをまだ持っていないキミ
アイ・ツーからお届けしちゃいマス!

チャンスです 逃がす手はない

■営業時間 AM11:00〜PM8:00

通販専用TEL.
06-633-9800

年中無休

Information & Interface
株式会社アイツー

大阪店/〒542 大阪市中央区難波千日前15-18

クリエイト特典

- ●全商品完全保証書付(メーカー保証)
- ●全国無料配達(一部離島の方は有料になります)
- ●配達日の指定OK(日曜・祭日にかかわらずお客様のご都合にあわせて配達します)
- ●どんな商品の組合せも自由自在(ご予算、用途に応じ自由自在にシステムアップできます)
- ●中古パソコン高額下取り(今お使いのパソコンをわずかな差額でグレードアップ)
- ●お支払い方法自由(低金利の均等払い、ボーナス一括払いもご利用ください)

営業時間(定休日▶渋谷店:日曜・祭日/横浜店:水曜)
AM10:00〜PM7:00

当社はX68000の販売認定店です。どんなことでも安心してご相談ください。

X68000 ビッグ・サマーセール開催中!

X68000 NEW PRO II

- ●CZ-653C(本体)……………………¥285,000
- ●CZ-603D(カラーディスプレイ)……¥ 84,800
- ●お好きなゲームソフト1本…………¥ 7,800
- ■定価合計…………………………¥377,600

クリエイト特価

均等払い	¥ 7,680×48回	¥ 9,890×36回	¥14,370×24回
ボーナス	なし	なし	なし

X68000 NEW EXPERT II

- ●CZ-603C(本体)……………………¥338,000
- ●CZ-613D(カラーディスプレイテレビ)……¥135,000
- ●CZ-8NJ2……………………………¥ 23,800
- ●お好きなゲームソフト1本…………¥ 9,800
- ■定価合計…………………………¥506,600

クリエイト特価

均等払い	¥ 9,970×48回	¥12,840×36回	¥18,660×24回
ボーナス	なし	なし	なし

X68000 EXPERT II HD

- ●CZ-613C(本体)……………………¥448,000
- ●CZ-604D(カラーディスプレイ)……¥ 94,800
- ●お好きなゲームソフト1本…………¥ 9,800
- ■定価合計…………………………¥552,600

クリエイト特価

均等払い	¥ 5,920×48回	¥ 7,400×36回	¥12,100×24回
ボーナス	¥30,000× 8回	¥40,000× 6回	¥50,000× 4回

X68000 SUPER HD

- ●CZ-623C-TN(本体・キーボード・マウス)………¥498,000
- ●CZ-613D-TN(カラーディスプレイ)……………¥135,000
- ●CZ-6BP1…………………………………………¥ 79,800
- ■定価合計…………………………………………¥712,800

クリエイト特価

均等払い	¥ 7,320×48回	¥10,100×36回	¥13,450×24回
ボーナス	¥42,000× 8回	¥50,000× 6回	¥80,000× 4回

※本広告に掲載の全商品の価格について消費税は含まれておりません。

X68000 NEW EXPERT II

ミュージシャンセット。これもTMネットワークだよ〜!

- ●CZ-603C………………………¥338,000
- ●CZ-605D………………………¥115,000
- ●MU1.B(MIDIボード&ソフト)……¥ 39,800
- ●CM32L…………………………¥ 69,000
- ●グラナダ………………………¥ 8,800
- ●JOYカード……………………¥ 1,800
- ■定価合計…………¥572,400▶超特価¥458,000

X68000 NEW PRO II

ゲーマーズセット。遊んで暮らせるSET!

- ●PRO II CZ653C……………¥285,000
- ●0.31CRT CZ603D…………¥ 84,800
- ●グラナダ……………………¥ 8,800
- ●Y'S…………………………¥ 8,700
- ●ポピュラス…………………¥ 9,800
- ●スーパーハングオン………¥ 8,800
- ●エージャックス……………¥ 8,800
- ●サーク………………………¥ 8,800
- ●アールタイプ………………¥ 7,800
- ●アナログJOYSTIC XE-1AP……¥ 13,800
- ■定価合計…………¥445,100▶超特価¥353,000

X68000シリーズ用 周辺機器・ソフト オール超特価!!

型番	品名	定価
CZ-6VT1	カラーイメージユニット	¥ 69,800
CZ-8NS1	カラーイメージスキャナ	¥188,000
CZ-6BE1A	IMB増設RAMボード	¥ 38,000
CZ-6BE2	2MB増設RAMボード	¥ 79,800
CZ-6BE4	4MB増設RAMボード	¥138,000
CZ-8NM3	マウス・トラックボール	¥ 9,800
BF-68PRO	高性能CRTフィルター	¥ 19,800
CZ-6BP1	数値演算プロセッサ・ボード	¥ 79,800
CZ-8NT1	トラックボール	¥ 13,800
CZ-6BM1	MIDIボード	¥ 26,800
CZ-8NJ2	アナログスティック	¥ 23,800
CZ-6TU	パソコンチューナ	¥ 33,100
SX-68M	MIDI I/F	¥ 19,800
XE-1AP	アナログジョイパッド	¥ 13,800

ソフト名	品名	定価
MUSIC PRO	MIDI版	¥ 28,800
MUSIC PRO-68K	マウスを使った楽譜ワープロ	¥ 18,800
SOUND PRO-68K	サウンドエディタ	¥ 15,800
Sampling PRO-68K	AD PCMサンプリングエディタ	¥ 17,800
Musicstudio PRO-68K V.1.1	MIDIマルチレコーディングソフト	¥ 28,800
OS-9/X68000	マルチタスクオペレーティングシステム	¥ 29,800
PRO-68K	サイバーノート	¥ 19,800
PRO-68K	ステーショナリー	¥ 14,800
Ccompiler PRO-68K	ソフト開発セット	¥ 39,800
Human 68K Ver2.0	開発ツールセット	¥ 9,800
PIO-6BE1-A	内蔵1MRAM	¥ 25,000
PIO-6BE2-2M	2MRAM	¥ 50,000
PIO-6BE4-4M	4MRAM	¥ 88,000
MU1-B	MIDI I/F＋ソフト	¥ 39,800

▲上記以外ビジネスソフト、最新ゲームソフト豊富に在庫あります。※送料はご注文の際お問合せください。 ●超特価販売中!

★この表以外の組合せ、お支払い方法もご自由にできます。
★X1シリーズ用、X68000シリーズ用各社ハードディスク/プリンタ等の周辺機器を大特価にて販売しております。
電話にてお問合せください。

●渋谷店

●横浜店

オール15%〜20%OFF

総合お問合せ先☎03-486-6541(代)

パソコン専門ショップ

ソフトクリエイト 渋谷/横浜

●渋谷店☎03-486-6541(代)
〒150:東京都渋谷区渋谷1-12-7 三和渋谷ビル
振込銀行:三井銀行 渋谷宮益坂支店(普)No.5000340

●横浜店☎045-314-4777(代)
〒221:横浜市神奈川区鶴屋町2-12-8 第1建設ビル
振込銀行:三和銀行 横浜駅前支店(普)No.310852

X68000 全機種取り揃え大特価セール

AIBIT アイビット電子株式会社

9月初旬 新店舗オープン!（京王線・北野駅前）

ビデオボード CZ-6BV1

¥21,000⇨**特価**

X68000に装着して、コンピュータ画像をビデオ信号として取り出せます。専用モニターがなくても、液晶ビジョンや大型テレビのビデオ入力端子につないで、コンピュータのゲームなどが楽しめます。

SHARP X68000

CZ-612C（本体）

プラス（ディスプレイ）組合せ

CZ-603D	特価¥358,000
CZ-602D	特価¥373,000
CZ-611D	特価¥383,000
CZ-613D	特価¥399,000

CZ-602C（本体）

プラス（ディスプレイ）組合せ

CZ-603D	特価¥287,000
CZ-602D	特価¥302,000
CZ-611D	特価¥312,000
CZ-613D	特価¥337,000

CZ-652C（本体）

プラス（ディスプレイ）組合せ

CZ-603D	特価¥255,000
CZ-602D	特価¥268,000
CZ-611D	特価¥280,000
CZ-612D	特価¥295,000

CZ-603C（本体）

プラス（ディスプレイ）組合せ

CZ-603D	特価¥300,000
CZ-602D	特価¥313,000
CZ-611D	特価¥323,000
CZ-613D	特価¥343,000

CZ-653C（本体）

プラス（ディスプレイ）組合せ

CZ-603D	特価¥287,000
CZ-602D	特価¥298,000
CZ-611D	特価¥310,000
CZ-612D	特価¥327,000

CZ-623C（本体）

プラス（ディスプレイ）組合せ

CZ-602D	特価¥473,000
CZ-603D	特価¥458,000
CZ-611	特価¥483,000
CZ-21HD	特価¥513,000

SHARP AX286N-H2 All in Note

定価¥398,000⇨**大特価!**

SHARP AX286L-F ラップトップ

定価¥428,000⇨**特価¥238,000**

NEC PC-9801n NOTE

定価¥248,000⇨**特価¥198,000**

TOSHIBA J3100SS DynaBook

定価¥198,000⇨**特価¥149,000**

《シャープ周辺機器(拡張、プリンター他)も常時取り扱っております。》

アイビット推奨ディスプレイ

●シャープCU-14FD
カラーディスプレイ
0.31チルト付A/24
定価¥74,800⇨
特価¥43,900

●シャープCZ-860D・BK
カラーディスプレイ
0.31チルト付A/D 15/24
定価¥92,200⇨
特価¥59,800

●三菱XC-1498CII
(14型アナログ)
ドットピッチ0.28
定価¥107,000⇨
特価¥59,800

XC-1498CII対応パソコン機種:PC-9801シリーズ/PC-286シリーズ/PC-386シリーズ/PC-8801シリーズ
(上記機種には付属の接続ケーブルで、接続可能)

●シャープCZ-830D・BK
(14型)
2モードオートスキャン方式
(アナログ/デジタル)
定価¥98,000⇨
特価¥54,800《在庫限り》

CZ-830D対応パソコン機種:CZ880C/881C。X1/TURBOシリーズ。ケーブルは本体付属を使用。
NEC PC-8801・9801シリーズ(XA・XLのみ不可)
MZ700/1500/2000/2200/2500各シリーズ(推奨品シャープ8D8K)。

●シャープCZ-602D-BK
(15型アナログTV/3モードオートスキャン)
定価¥99,800⇨
特価¥75,000

CZ-602D対応パソコン機種:※X1シリーズ/※X1 turboシリーズ/X1 yurboZシリーズ/X68000シリーズ/PC8801シリーズ/PC-9801シリーズ/PC-286シリーズ
(※は接続ケーブルAN1506が必要です)

●シャープ CZ-603D-GY・BK
(15型カラーディスプレイ)
ドットピッチ3.9
定価¥84,800⇨
特価

CZ-603D対応パソコン機種：※X1シリーズ/※X1 turboシリーズ/X1 yurboZシリーズ/X68000シリーズ/PC8801シリーズ/PC-9801シリーズ/PC-286シリーズ
(※は接続ケーブルAN1506が必要です)

各種特選・特価品!

パソコンファクス MZ-1V01

"プリンタ・コピー・ファクス"
1台3役のスグレモノ
限定セット販売!

●MZ25セット(インターフェースソフト付)
標準価格合計¥342,800➡**¥120,000**
●MZ-1V01(本体のみ)
標準価格合計¥278,000➡**¥ 98,000**

ハガキもOK、New MZプリンタ

漢字カラー熱転写プリンタ
シャープMZ-1P22

<24×24ドット漢字●7色カラー●漢字30字/秒高速印字●MZ1P17とフルコンパチ●5KBのバッハメモリ付>適応パソコン:MZ2000、2500、5500、6500シリーズ、X1シリーズ、X68000シリーズ他

標準価格¥59,800
➡**特価¥25,000**

《在庫限り》PC-E500PJ

定価¥28,800➡**大特価**
SHARPポケコン全機種、Z80ボード他、太平洋工業製品全機種取扱
●PC-E500PJご購入の方に、もれなく「ポケコンジャーナル特別号」を進呈!

PC-500と各種パソコンをつなぐインターフェースケーブル
CE-140T ¥8,800

キヤノン LASER SHOT プリンター LBP-B406S

定価¥498,000⇨**特価!**

キヤノン LASER SHOT プリンター LBP-A404S

定価¥265,000⇨**特価!**

シャープMZ-1X30 モデムホン

(1×19上位機種)《在庫限り》

<300/1200BPS全2重通信対応モデム内蔵●音声入出力端子付●ダイヤルパルス/プッシュボタン対応●プッシュボタン音解析機能●シャープ手順、CCITT、V25bis通信手順サポート>

標準価格¥98,000➡**¥29,800**

FM TOWNS お買い得セット

1. TOWNS-1	(本体)	¥338,000
2. FMT-ME(1M)	(増設メモリー)	¥ 60,000
3. FMD-FD301	(増設FDユニット)	¥ 28,000
4. FMT-KB101	(キーボード)	¥ 20,000
5. FMT-DP531	(カラーディスプレイ)	¥ 89,800
6. TOWNS-OS V1.1 L20		¥ 20,000
	定価合計	¥555,800

大特価!¥285,000

〈全商品新品完全保証付〉■シャープポケコン全商品販売中。カタログ、特価表ご請求ください(〒72)

☎0426-45-3001~3

FAX.0426-44-6002

●営業時間/10:00~19:00●電話受付/20:00迄可●定休日/日曜日(祭日営業)

SHARP SUPER XEX SHOP

アイビット電子株式会社 〒192 東京都八王子市北野町560-5

上記の広告商品はすべて店頭販売もしております。

全国通信販売
北海道から沖縄まで

★送料はご注文の際にお問い合わせ下さい。
★掲載の商品は、すべて新品、保証書付きです。
★掲載の商品は充分用意してありますが、ご注文の際は、在庫の確認の上、現金書留または、銀行振込でお申し込み下さい。全商品クレジットでも扱っております。
★お申し込みの際は必ず電話番号を明記して下さい。
★商品、品切れの節はご容赦下さい。

富士銀行八王子支店 (普)1752505

●本誌発売時には上記価格よりさらにお求めやすい価格に変更されている場合があります。●この広告の商品にはすべて送料・消費税は含まれておりません。

ACCESS

X1エミュレータ

好評発売中

定価¥9,800

X1エミュレータはX68000上でX1シリーズのアプリケーションを実行するためのソフトエミュレータです。X1のアプリケーションを完全にソフトウェアのみでエミュレートしているため、X1上での実行速度と比較して、平均3〜5倍程度おそくなりますが、X68000のマシン上に実現した仮想X1マシンを楽しめます。また、X1とX68000の相互間でファイルを転送するためのユーティリティと専用ケーブルが付属しますので、X1上で作り上げたソフトの資産をX68000上に移行することも簡単にできます。

X1エミュレータの機能

- X1エミュレータはX1に相当する機能をエミュレート。この仮想コンピュータには最大4つのドライブが仮想的に接続。
- X1エミュレータからみたドライブはHuman68kのドライブ上にあるファイルで仮想的に実現。このファイルはX1用の5" 2Dディスクのイメージをファイル転送ユーティリティでまるごと転送したもの。
- X1エミュレータで仮想的に実現したX1は仮想ドライブから起動。このため仮想ドライブ用ファイルには、X1を立ち上げるために必要なHuBASICやCP/Mなどのシステムプログラムが必要。
- X1エミュレータでは、X1の持つVRAMを含むメモリイメージやZ80CPUを仮想的にソフトウェアで実現。

ファイル転送ユーティリティ

ディスク転送

X1ディスク⟷X68000 Human68k(5" 2Dディスクイメージファイル)

- X1エミュレータではHuman68k上のディスクイメージファイルを仮想ドライブとして使用。

ファイル転送

X1 BASIC:CP/M⟷X68000 Human68k

- X1で作ったプログラム&データをX68000上で使用。

※付属の専用ケーブルをX1とX68000に接続してファイルを転送します。

X1エミュレータQ&A

Q. ファイル転送のために別途RS-232Cケーブルを買わないといけないのですか?

A. 専用のケーブルが付属しますのでその必要はありません。

Q. X1BASICのプログラムをX68000上のX-BASICで使えますか?

A. 通常のセーブではコードが違うので使用できませんが、アスキーセーブしたファイルであればX-BASIC上でそのままロード可能です。

Q. TurboBASICで作成した住所録などの漢字を含んだデータがあるのですがX68000上にファイル転送できますか?

A. X1TurboもX68000も漢字はシフトJISコードなのでファイルの転送は可能です。ただし、漢字ROMを必要とするものはサポートしていません。

Q. Turbo用のソフトは動きますか?

A. X1用のみでTurbo専用のソフトは動きません。

Q. ゲームは動きますか?

A. 純粋にBASICでかかれたものは動きますが、プロテクトがかかったものや直接ハードをアクセスするような市販のゲームは動きません。

*タイミング等ハードウェアに依存するようなソフトは、原理上実行できない、もしくは正常に動作しない場合がありますのでご注意ください。
*一部サポートしていない機能があります。

X1エミュレータ通信販売 購入希望として住所、氏名、電話番号をお知らせください。注文書をお送り致します。

*この商品価格には消費税は含まれておりません。
*CP/Mはデジタルリサーチ社の商標です。
文中のソフトウェアは各社の商標です。
*製品の仕様、名称は予告なく変更する場合がございますのであらかじめご了承ください。

有限会社アクセス 〒101 東京都千代田区神田神保町1-64 神保町協和ビル7F
☎03(233)0200(代) FAX.03(291)7019

資料請求券 oh!X 9月号

パソコン/ワープロ通信ネットワークサービス

J&P HOT LINE

SIG探訪PART 3

んな人、ぜひおいで〜
の好きな人・花の好きな人
樹の好きな人・鳥の好きな人
昆虫の好きな人・山歩きの好きな人
水辺の好きな人・地球が好きな人
大自然の大好きな人

NATURAL SPACE (ジャンプコード:NATURAL)

SIGの御意見番 Subop:ばあど 野鳥担当

あゆ: 目下自然よりも女性に興味がある

OFF-op:豆腐 「花鳥風月」編集者

Ueda: ワーコムネットのSIGOP

Subop:MAT コンピュータ担当

カイエネットのメンバー

SIGOP:DEW 植物担当

ミオン: 詩情あふれるメッセージでファンが多い。

カイエ: カイエネットのSYSOP

兵庫県は六甲山の山の中で、ホタル・ミーティングをやりました。ホタルを初めて見る人もあって、雨の中のたった2匹のホタルにみんな感動しました。そのあと小さな水溜まりでイモリの親子やオタマジャクシやヒルやミズスマシと遊びました。真っ暗闇の山中で懐中電灯片手にイモリをおっかけるいい歳をした人々のようすに、はじめて参加した若い人たちが呆れていました。

自然誌「花鳥風月」

NATURAL SPACEの守備範囲を遥かに超えた広い視野で編集されています。読んでみたい方は、SIGまでどうぞ。そうして気に入ったら執筆同人になってください。

21世紀は「自然の世紀」!
あつまれナチュラリスト(大自然探険隊)

今ほど自然を痛めつけている時代はないなぁというのが実感。でも、自然大好きのワタシたちは嘆いてばかりではありません。牛乳パックを材料に**手漉きのハガキを作る人**(SIGメンバーに送られたハガキはすこし分厚くてザラザラしていましたが、地球への愛情がいっぱいの、すごく感動的なものでした。)、**休みのたびに水辺の鳥を観にいく人、資源のリサイクルに取り組む人**……メンバーの一人ひとりが身近なところや、あるいは個人の力ではどうしようもないほどデッカイ問題に、コツコツと取り組んでいます。SIGのボードでは、**自然を絵で語るために作った「YUKI画像通信システム」**による美しい画像通信が自慢です。(現在、X68000と98シリーズでデータ互換できます。)

その他楽しいメニューがまだまだいっぱい!

- ★J&Pならではのパソコン・家電製品の会員割引もある**ONLINE SHOPPING。**
- ★J&Pだから強い!!パソコン情報をはじめとする役に立つ**DATA BASE。**
- ★みんなでおしゃべり**オンライントーク**(CHAT機能)。
- ★地域別・テーマ別ボードで充実の**BBS**(電子掲示板)。
- ★ビジュアルデータもばっちり送受信できる**X-MODEM。**

J&P HOT LINEへのご入会はスタータキットで。

お求めは、下記のお店へ。又は現金書留にて、¥3,000+¥90(消費税3%)=¥3,090を事務局までお送り下さい。
すぐにスタータキットをお送りします。

お問い合わせは 〒556 大阪市浪速区日本橋西1-6-5 上新電機株式会社
J&P HOT LINE事務局宛 TEL.(06)632-2521

スタータキットのお求めはJ&P各店でどうぞ。

店名	住所	電話
渋谷店	東京都渋谷区道玄坂2丁目28番4号	☎(03) 496-4141
町田店	東京都町田市森野1丁目39番16号	☎(0427)23-1313
八王子店	東京都八王子市旭町1番1号八王子そごう7F	☎(0426)26-4141
立川店	東京都立川市幸町4-39-1	☎(0425)36-4141
本厚木店	厚木市中町3-4-3	☎(0462)25-1548
富山店	富山市桜町2-1-10	☎(0764)32-3133
金沢店	金沢市入江2-63	☎(0762)91-1130
寺地店	金沢市寺地2-3	☎(0762)47-2524
大須店	名古屋市中区大須4丁目2-48	☎(052)262-1141
テクノランド	大阪市浪速区日本橋5丁目6番7号	☎(06) 634-1211
メディアランド	大阪市浪速区日本橋5丁目8番26号	☎(06) 634-1511
コスモランド	大阪市浪速区難波中2丁目1番17号	☎(06) 634-3111
U.S.LAND	大阪市浪速区日本橋4丁目9番15号	☎(06) 634-1411
ビジネスランド	大阪市北区梅田1-1-3大阪駅前第3ビルB2	☎(06) 348-1881
梅田店	大阪市北区小松原町1-10	☎(06) 362-1141
高槻店	高槻市高槻町11番16号	☎(0726)85-1212
くずは店	枚方市楠葉花園町15番2号	☎(0720)56-8181
千里中央店	豊中市新千里東町1-3SENCHU PAL 2番街4F	☎(06) 834-4141
摂津富田店	高槻市大畑町24-10	☎(0726)93-7521
寝屋川店	寝屋川市緑町4-20	☎(0720)34-1166
藤井寺店	藤井寺市岡2丁目1番33号	☎(0729)38-2111
岸和田店	岸和田市土生町2451-3	☎(0724)37-1021
さんのみや1ばん館	神戸市中央区八幡通3-2-16	☎(078)231-2111
西宮店	兵庫県西宮市河原町5-11	☎(0798)71-1171
姫路店	姫路市東延末1丁目1番住友生命姫路南ビル1F	☎(0792)22-1221
京都寺町店	京都市下京区寺町通仏光寺下ル恵美須之町549	☎(075)341-3571
京都近鉄店	京都市下京区烏丸通七条下ル東塩小路町702	☎(075)341-5769
和歌山店	和歌山市元寺町4丁目4番地	☎(0734)28-1441
奈良1ばん館	奈良市三条町478-1	☎(0742)27-1111
郡山インター店	大和郡山市横田693-1	☎(07435)9-2221
熊本店	熊本市手取本町4-12	☎(096)359-7800

T4910217909569 雑誌 02179-9